

**A PROJECT REPORT ON**  
**G - VAAHAN**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ANUDEV K V**

**REG.NO: DB20BCAR22**

**VAISHNAV C V**

**REG.NO: DB20BCAR13**

**DEON SHAJI**

**REG.NO: DB20BCAR28**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**

**A PROJECT REPORT ON**  
**G - VAAHAN**  
**CONNECTION OF AMBULANCE DRIVERS**  
**AND HOSPITALS IN AN LOCALITY**

Submitted in partial fulfilment of the requirement for award of the degree Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**VAISHNAV C V**

**REG.NO: DB20BCAR13**

**ANUDEV K V**

**REG.NO: DB20BCAR22**

**DEON SHAJI**

**REG.NO: DB20BCAR28**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**



**DON BOSCO ARTS & SCIENCE COLLEGE**  
**ANGADIKADAVU**  
**IRITTY, KANNUR**



**CERTIFICATE**

*Certified that this report titled **G-VAAHAN -CONNECTION OF AMBULANCE DRIVERS AND HOSPITALS IN AN LOCALITY** is a bonafide record of the project work done by **Anudev K V (Reg no. DB20BCAR22)** , **Vaishnav C V (Reg no. DB20BCAR13)** and **Deon Shaji (Reg no. DB20BCAR28)** under the supervision and guidance ,towards partial fulfilment of the requirement for award of the degree of bachelor of computer application (BCA) of the Kannur university.*

Project Guide

Head of the Department

Angadikadavu

External Examiner

Date:

1.

2.

# Declaration

We ANUDEV K V , VAISHNAV C V and DEON SHAJI, sixth semester BCA student of Don Bosco Arts & Science College, Angadikadavu, under Kannur University do hereby declare that the project entitled **“G-VAAHAN - CONNECTION OF AMBULANCE DRIVERS AND HOSPITALS ”** is the original work carried out by me in the sixth semester under the supervision of Mrs. Sindhu Titus, HOD of the Department of BCA, Don Bosco Arts & Science College, Angadikadavu, in partial fulfilment of the requirement for the award of the degree Bachelor of Computer Application, Kannur University.

Angadikadavu

Date

ANUDEV K V

DEON SHAJI

VAISHNAV C V

# **ACKNOWLEDGEMENT**

First of all we thank the lord almighty for his immense grace and blessings showered on me at every stages of this work. I am greatly indebted to our Principal Fr. Dr. Francis Karackat SDB, Don Bosco Arts & Science College, Angadikadavu for providing the opportunity to take up this project as part of my curriculum.

We deeply indebted to my project guide Mrs. Sindhu Titus , HOD of department of BCA, for her assistance and valuable suggestions as guide. She made this project a reality.

We express our sincere thanks to Mr. Hebin Layola, Mrs. Sruthi, Mrs. Fincy Cyriac and Mrs. Vineetha Mathew, lecturers of department of BCA, for their valuable suggestions during the course of this project. Their critical suggestions helped me to improve the project work.

Acknowledging the efforts of everyone, their chivalrous help in the course of the project preparation and their willingness to collaborate with the work, their magnanimity through lucid technical details lead to the successful completion of my project.

We would like to express my sincere thanks to all my friends, colleagues, parents and all those who have directly or indirectly assisted during this work.

# CONTENTS

<b>chapters</b>	<b>contents</b>	<b>Page No</b>
1	INTRODUCTION	1
2	SYSTEM ANALYSIS	7
2.1	Existing System	8
2.1.1	Disadvantage Of Existing System	8
2.2	Proposed System	8
2.3	Feasibility Study	9
2.3.1	Economic Feasibility	9
2.3.2	Technical Feasibility	10
2.3.3	Behavioral Feasibility	10
2.4	System Specification	11
2.4.1	Software Specification	11
2.4.2	Hardware Specification	11
2.5	Identification Of Actors	12
2.6	Identification Of Use Cases	13
2.6.1	Use Cases For The Actor Admin	13
2.6.2	Use Cases For The Actor Ambulance Driver	14
2.6.3	Use Cases For The Actor User	15
2.6.4	Use Cases For The Actor Hospital	15
2.6.5	Use Case Diagram	17

3	SYSTEM DESIGN	21
3.1	Introduction	22
3.2	Database Design	22
3.3	Table Design	23
3.4	Data Flow Diagram	29
3.5	ER Diagram	39
4	CODING	41
4.1	Input Interface	42
4.2	Output Interface	42
4.3	Software Description	42
4.4	HTML	42
4.4.1	CSS	45
4.4.2	JavaScript	48
4.4.3	MySQL	50
4.4.4	Python	53
4.4.5	Flask	54
5	CODING PAGES	55
6	System Testing	62

6.1	Testing And Evaluation	63
6.2	Testing Strategies	64
6.3	Testing Techniques	65
6.3.1	White Box Testing	65
6.3.2	Black Box Testing	66
6.3.3	Unit Testing	66
6.3.4	Integration Testing	67
6.3.5	Acceptance Testing	67
6.3.6	Output Testing	68
7	SYSTEM IMPLEMENTATION AND DEPLOYMENT	69
8	CONCLUSION	71
9	REFERANCE	73
10	APPENDIX	75

# **1.INTRODUCTION**

## **1.1 Project Overview**

By means of G - Vaahan we are planning to organize the ambulance drivers of a locality so that the users can interact with them in case of emergencies. Users can have a permanent account with registration details like name , phone number, E- mail, Id proof, password . Immediate (Emergency) users can login with an OTP provided by the admin. Drivers should register in the software with their photo proof , name, phone number, license proof , ambulance no, password . Hospitals nearby are also linked within the application

## **NEED FOR THE SYSTEM**

Proper treatment can be ensured in case of emergencies. Within a minute ambulance and hospital services are provided to the users. Users can interact with the application in case of emergencies without a user account. Ambulance drivers of a locality is organized in G-Vaahan. In case of emergencies hospitals are able to do preparations for admitting the patients. Administrator has the overall control of application

## **OBJECTIVES AND SCOPE**

The main objectives behind the development of this project are:

1. Serving society by saving life.
2. Organizing ambulance drivers
3. Making activities of application secure under supervision of an administrator.
4. In case of Emergency ambulance service can be ensured.

The scope of this project is high as it reduces manual work and utilises the resources in an efficient manner with minimum usage of time.

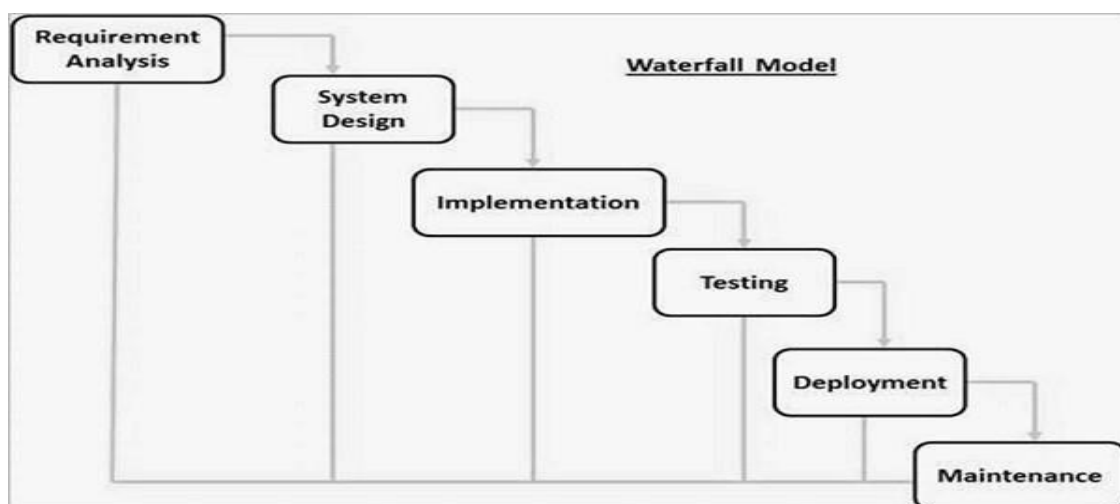


## MODEL:

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially

### Waterfall Model - Design:

In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. Following is the pictorial representation of Iterative and Incremental model:



The sequential phases in Waterfall model are –

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

### **Waterfall Model - Application:**

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are –

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

### **The advantages of the waterfall model SDLC Model are as follows:**

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Some of the major advantages of the Waterfall Model are as follows –

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

## **The disadvantages of the waterfall model SDLC Model are as follows:**

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows –

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

## **2. SYSTEM ANALYSIS**

## **Introduction**

System analysis is the process of collecting and interpreting facts, understanding problems and using the information to suggest improvement on the system. This will help to understand the existing system and determine how computers make its operation more effective. The aim of this analysis is to collect detailed information on the system and the feasibility study of the proposed system.

## **2.1 EXISTING SYSTEM**

Emergency Rescue Ambulance Driver – “Ambulancee.com.”

### **2.1.1 The Disadvantages of Existing System**

1. These type of available apps are not working properly.
2. Hospitals are not linked with in this application.
3. In case of emergencies users cannot use this application without registration.

## **2.2 PROPOSED SYSTEM**

Since hospitals and ambulance are not connected , we are trying to reduce difficulties caused by the patients , thereby making the ambulance driver’s job easier.In the proposed system of organizing the ambulance drivers of a locality , we are building a model , so that the users can interact with them in case of emergencies . To develop such a model we are using a dataset of ambulance driver’s details like photo proof , name, phone number ,place ,license proof ,vehicle details . The project focuses on interconnecting hospitals and ambulance drivers within a locality.

## **2.3 Feasibility Study**

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioral feasibilities.

The proposed system is theoretically investigated to check the feasibility and found that they are more reliable and efficient in the cases given below. There are three aspects in the feasibility study portion of the preliminary investigation.

✓ Economic feasibility

✓ Technical feasibility

✓ Behavioral feasibility

The proposed system must be evaluated from a technical point of view first, and if technical feasible their impact on the organization must be assessed. If compatible, the operational system can be devised. Then they must be tested for economic feasibility.

### **2.3.1 Economic Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors which affect the development of a new system is the cost it would require. Since the system

developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

### **2.3.2 Technical Feasibility**

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, procedures and staff. Having identified an outline system, the investigation must go on suggest the type of equipment, required method developing the system, of running the system once it has been designed. The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Though the technology become obsolete after some period of time, due to the fact that newer version of some software supports older versions, the system still be used. So there are only minimal constraints involved with this project. The system has been developed using C# and .NET, along with the database software SQL server, thus we could conclude that the project is technically feasible for development.

### **2.3.3 Behavioral Feasibility**

People are inherently resistant to change and computers have been known to facilitate change. The System is designed in user friendly manner and we need to provide any special training for the persons using this software. The operating system used is Windows 10, which is also user friendly. Since the application is web biased and can easily accessed in a web browser, which is quite familiar to the intended users, it does not have any operational barriers. So no need to provide any special training for using this application software and hence it is behaviourally feasible.



## **2.4 System Specifications**

System Specification deals with the technical aspects the project has to meet in minimum to work successfully. This also includes the different aspects the software requirement is determined from. The technical details typically include:

- Software Specification
- Hardware Specification

### **2.4.1 Software Specifications**

The software required for the application depends on the following factors:

- ✓ The flexibility of the software
- ✓ Software contracts
- ✓ Limitation of the software

### **2.4.2 Software Requirement**

One of the most difficult task is selecting software for the system, once the system requirements is found out then we have to determine whether a particular software package fits for those system requirements. The application requirement:

- Front end : HTML,XML
- Back end : MySQL
- Operating system : windows 7 or above
- Language : Python,Java
- Software : Pycharm,Android Studio

### **2.4.3 Hardware Specifications**

The software required for the application depends on the following factors:

- ✓ Determining size and capacity requirements.
- ✓ Computer evaluation and measurement.
- ✓ Financial factors.
- ✓ Maintenance and support.

#### **2.4.4 Hardware Requirement**

The selection of hardware is very important in the existence and proper working of any software. Then selection hardware, the size and capacity requirements are also important

- Processor : Intel Pentium Core i3 and above
- Primary Memory : 4GB RAM and above
- Storage : 320 GB hard disk and above
- Display : VGA Color Monitor
- Key Board : Windows compatible
- Mouse : Windows compatible

### **2.5 Identification of Actors**

A use cases represents the functionality of an actor. It is defined as a set of actions performed by a system, which yields an observable result. An ellipse containing its name inside the ellipse or below it represents. it. It is placed inside the system boundary and connected to an actor with an association. This shoes how the use cases and the actor interact.

We can identify the actors through a list of questions. The answers to these questions bring out the actors of the system is.

- **Admin**
- **User**
- **Ambulance driver**
- **Hospital**

Here we need to specify the use cases of each actor.

## **2.6 Identification of use cases**

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yield an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

To find out the use cases, ask the following questions to each of the actors.

- ✓ Which functions does the actor require from the system? What does the actor need to do?
- ✓ Does the actor need to read, create, destroy, modify or store some kind of information in the system?
- ✓ Does the actor have to calculate something? And want to provide information for others?
- ✓ Could the actor's daily work be simplified or made more efficient by adding new functions to the system (typically functions which are currently not automated in the system)?

### **2.6.1 Use cases for the actor Admin**

#### **Login:**

The first step involved is login. The admin can login to the website using username and password.

#### **Verify Hospital :**

Admin can verify and approve Hospitals .

#### **Verify Driver :**

Admin can verify and approve Driver.

**Block driver:**

Admin can block driver.

**View User :**

Admin can view User and verify

**Block User:**

Admin can block User if found suspicious .

**View Feedback :**

Admin can view Feedbacks from users.

**2.6.2 Use cases for the actor Ambulance Driver****Registration :**

Driver can Register with license details, Id proof, contact details etc.

**Login:**

Driver can login using username and password.

**Update location :**

Driver can update location.

**View Emergency Request :**

Driver can view Emergency Request.

**Accept Request:**

Driver can accept Request from users.

**Notify Nearest Hospital:**

Driver can notify nearest hospital

**Track Request Location :**

Driver can track request location.

**Report user :**

Driver can report suspicious user.

**View feedback :**

Driver can view user's feedback

**2.6.3 Use cases for the actor User****Registration :**

User can register using name, contact details, E mail, Id proof.

**Login:**

User can login using username and password.

**Track Ambulance :**

User can track ambulance driver.

**Notify nearest hospital :**

User can notify nearest hospital .

**Accident detection (Threshold):**

Notifies ambulance driver and hospital by Threshold detection.

**Send emergency feedback :**

User can send emergency feedback.

**Send feedback :**

User can send feedback.

**2.6.4 Use cases for the actor Hospital****Registration:**

Hospitals can register with hospital name, hospital contact, hospital place, hospital pin, hospital post.

**Login :**

Hospitals can login with Hospital name and password.

**View Notification:**

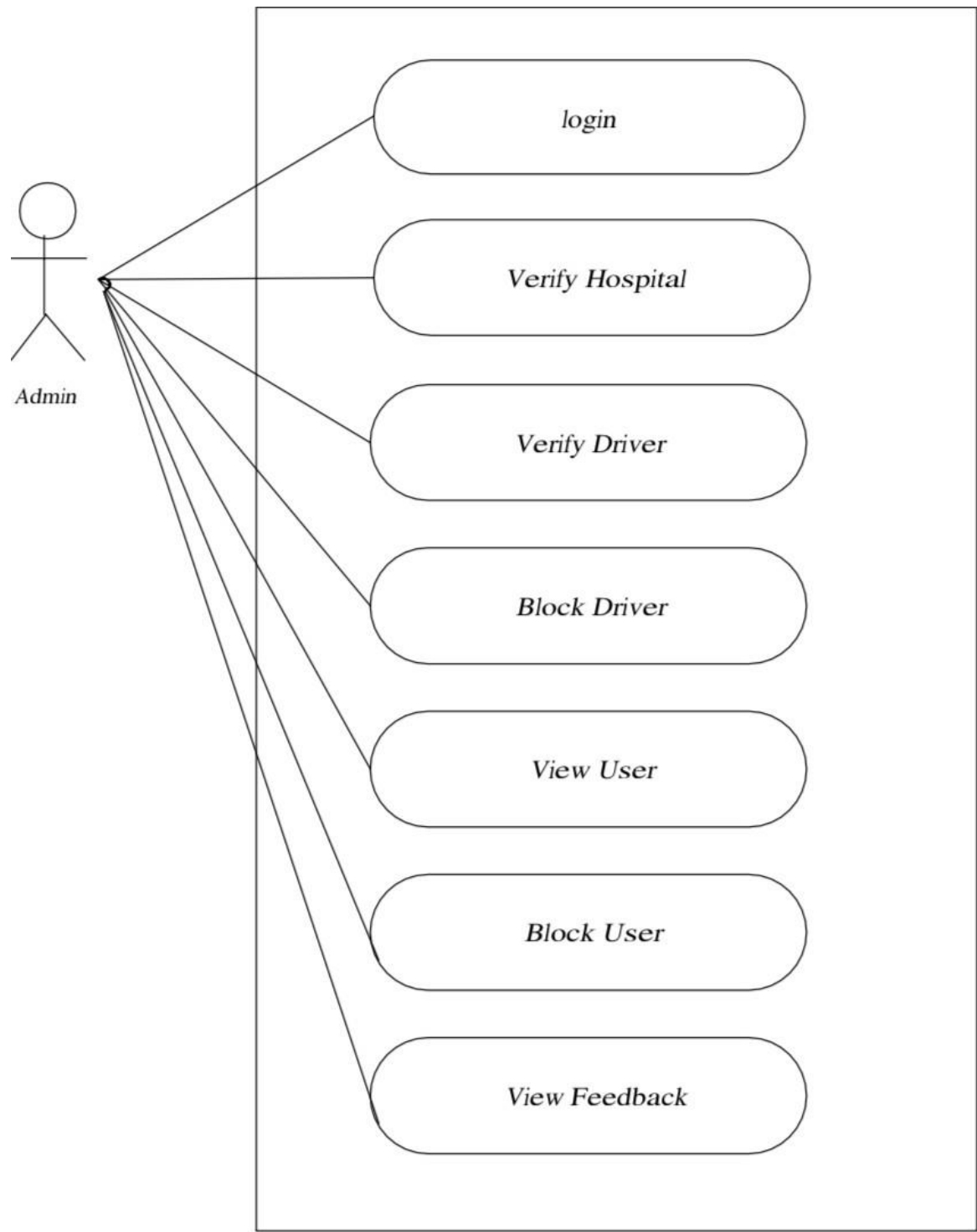
Hospitals can view notifications from users.

**Track Ambulance :**

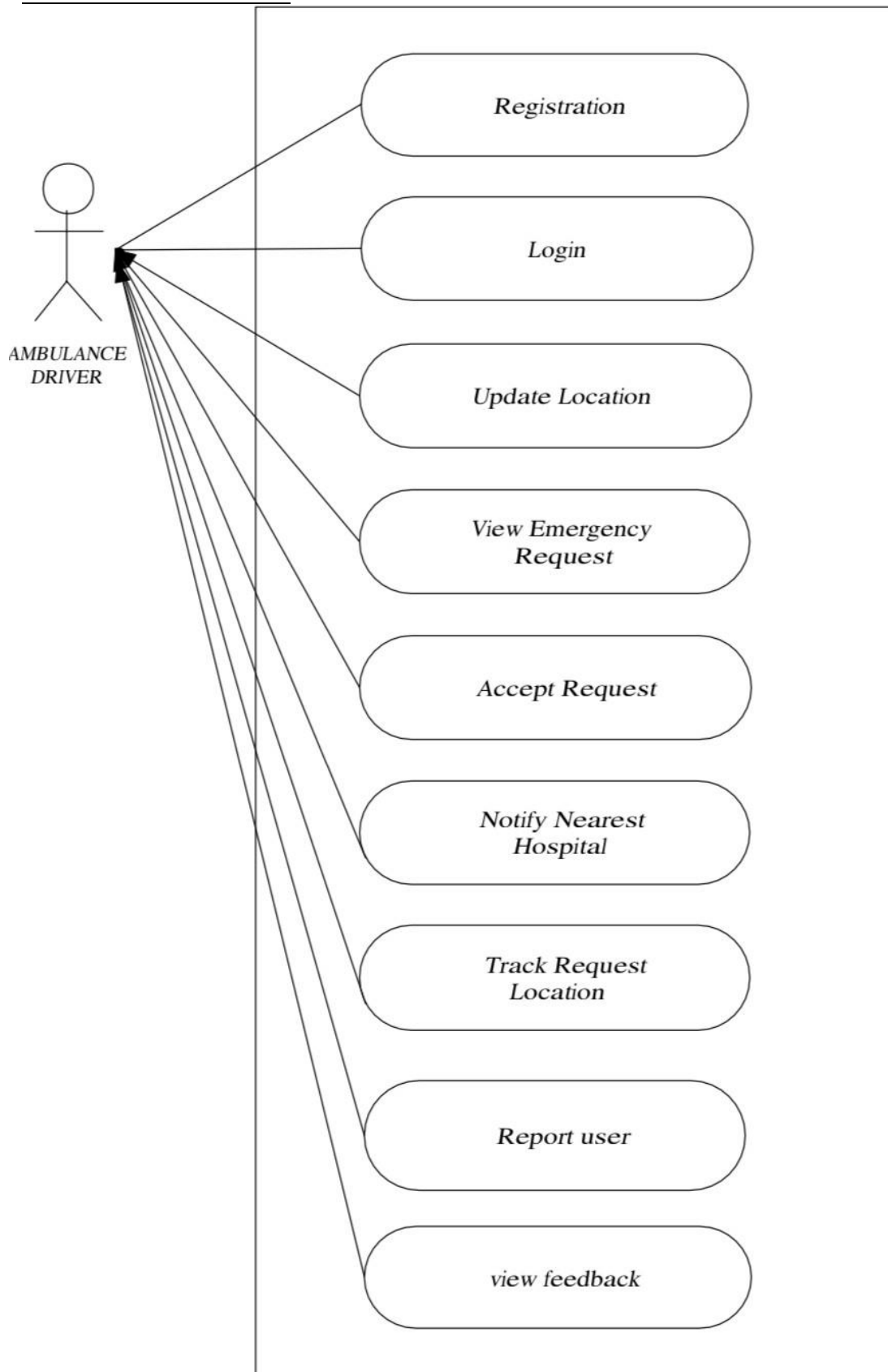
Hospital can track ambulance driver.

2.6.5 USE CASE DIAGRAM

ADMIN

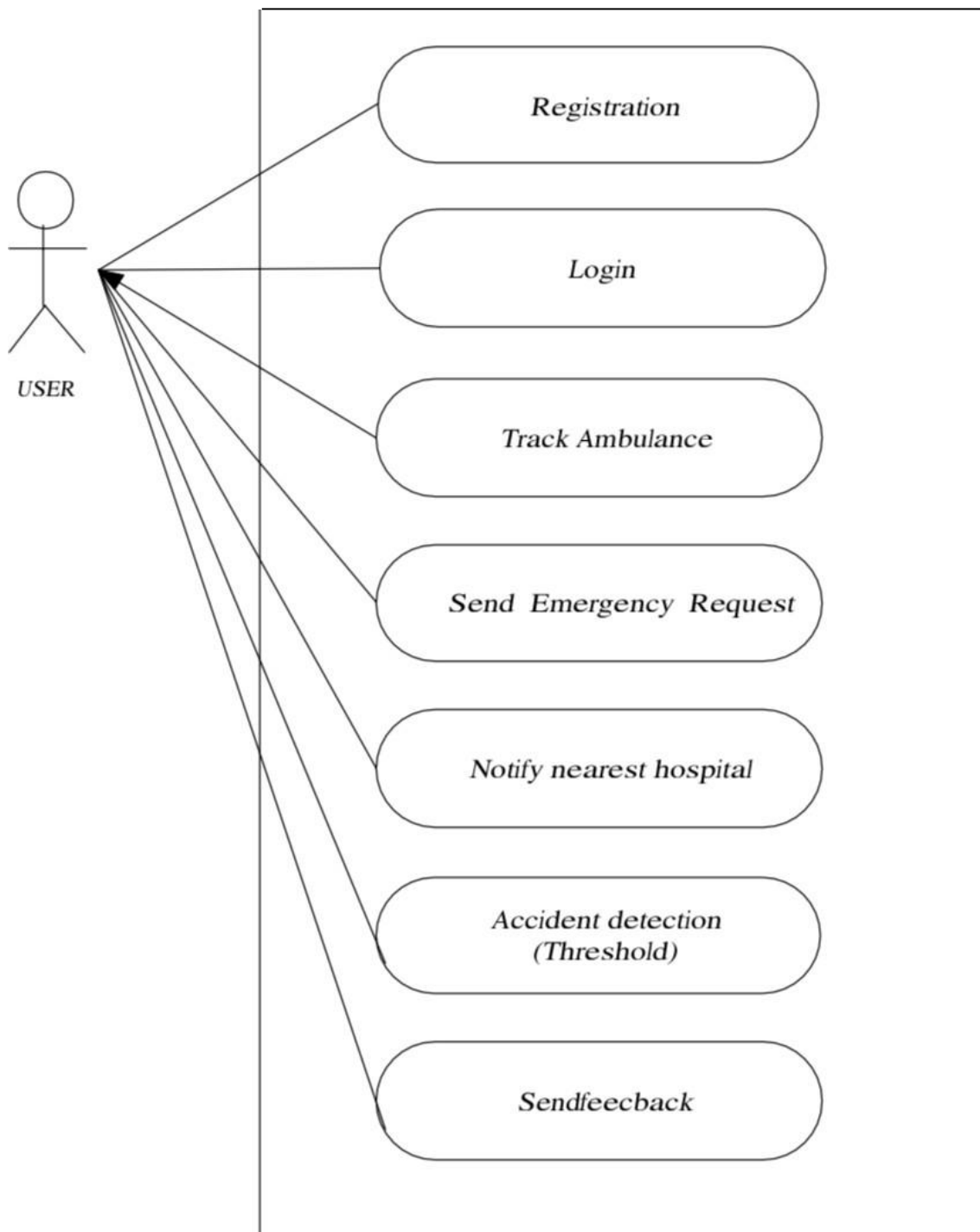


## AMBULANCE DRIVER

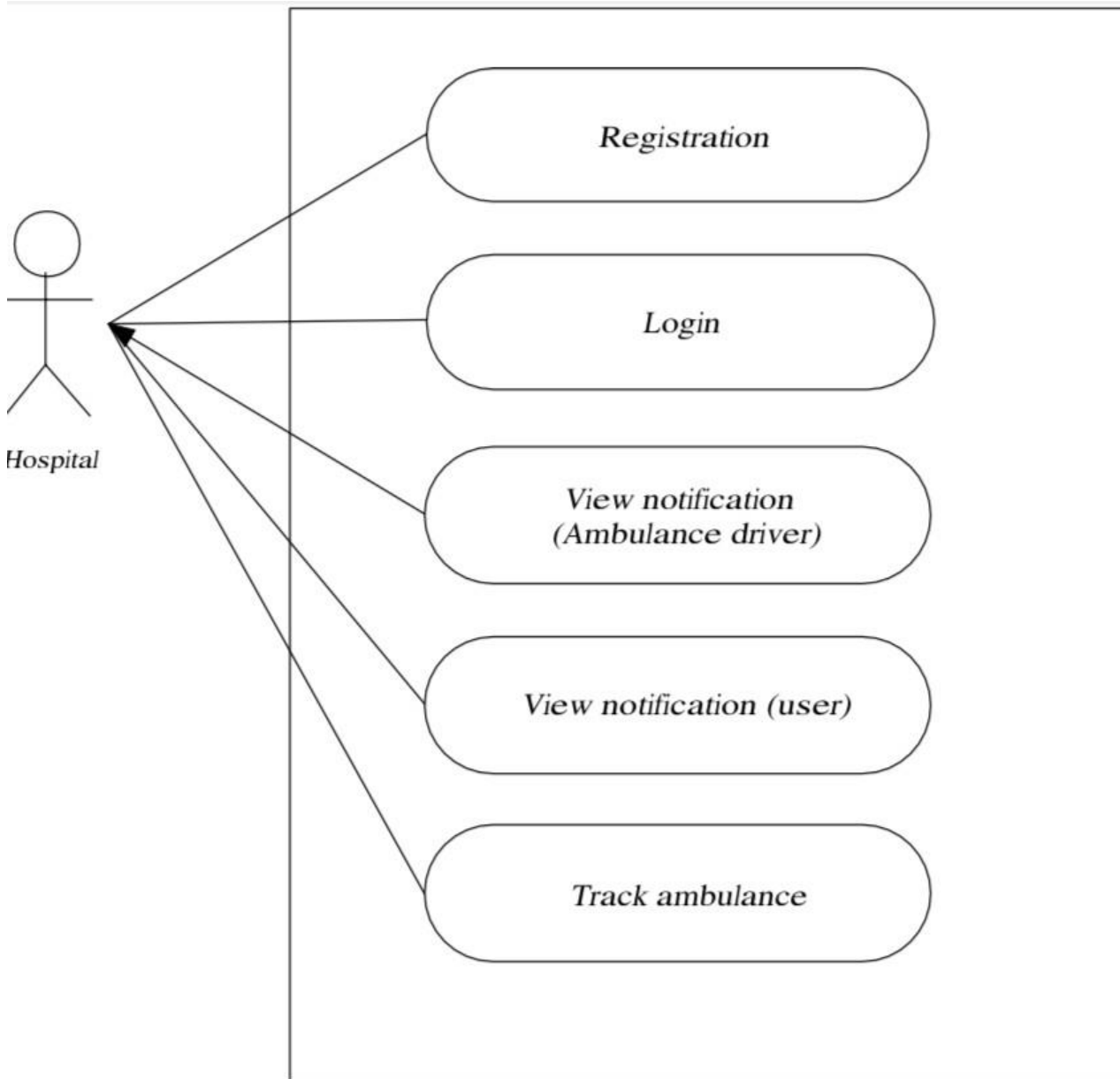




## USER



## HOSPITAL



### **3. SYSTEM DESIGN**

### **3.1 INTRODUCTION:**

System design provides an understanding of the procedural details, necessary implementing the system recommended in the feasibility study. Basically it is all about the creation of a new system. This is a critical phase since it decides the quality of the system and has a major impact on the testing and implementation phases.

**System design consists of three major steps.**

- Drawing of the expanded system data flow charts to identify all the processing functions required.
- The allocation of the equipment and the software to be used.
- The identification of the test requirements for the system.

### **CHARACTERS OF DESIGN**

- A design should exhibit a hierarchical organization that makes intelligent use of control among components of the software.
- A design should be modular that is, the software should be logical.
- A design should contain distinct and separable representation of data and procedure.
- A design should lead to interface that reduce the complexity of the connections between modules and with the external environment.

### **3.2 Database Design**

A Database is a collection of inter related data stored with minimum redundancy to serve many users quickly and efficiently. In database design data independence, accuracy, privacy

and security are given higher priority. Database design is an integrated approach to the file design. This activity deals with the design of the physical data base. All entities and attributes have been identified while creating the database. The database design deals with the grouping of data into number of tables so as to.

- ✓ Reduplication of data.
- ✓ Minimize storage space.
- ✓ Retrieve the data efficiently.

Following are some guidelines for the database design:

- Design a relational schema so that it is easy to explain its meaning. Do not combine attributes from multiple entry and relationship type into a single relation.
- Design the database schema so that no insertion, deletion or modification anomalies are present in the relation.
- As far as possible, avoid placing attributes in the base relation whose values may frequently be null.
- Design relation schema so that they can be joined with equality conditions on attributes that are either primary keys or foreign keys in a way that no spurious tuples are generated.

### **3.3 Table Design**

DB design is required to manage large bodies of information. The management of data involves both the definition of the structure of storage of information and provisions of mechanism for the manipulation of information. For developing an efficient database certain conditions have to be fulfilled such as:

- Control Redundancy
- Ease of Use

- Data Independence
- Accuracy and Integrity

There are five major steps in design process:

- Identify the table and relationship.
- Identify the data that is needed for each table and relationship.
- Resolve the relationship.
- Verify the design.
- Implement the design

The Database Consist of the following tables given below.

### 3.4 G-VAAHAN

#### *1.Login table*

Column name	Data type	Constraints	Description
login_id	int	primary key	unique identifier
username	varchar(50)	not null	name of user
password	varchar(50)	not null	secret key
usertype	varchar(50)	not null	To specify the role

#### *2.User table*

Column name	Data type	Constraints	Description
User_id	int	primary key	Unique identifier
Username	varchar(50)	not null	Name of user
Usercontact	varchar(50)	not null	Contact of user
Useremail	varchar(50)	not null	Email of user
Userphoto	Varchar(100)	Not null	Photo of the user

### *3. accident table*

Column name	Data type	Constraints	Description
accident_id	int	primary key	unique identifier
user_id	int	not null	user identifier
latitude	varchar(50)	not null	
longitude	varchar(50)	not null	
date	varchar(50)	not null	
time	varchar(50)	not null	
place	varchar(50)	not null	

### *4. driver table*

Column name	Data type	Constraints	Description
Driver_id	int	primary key	unique identifier
Drivername	varchar(50)	not null	
Driverlicense	varchar(50)	not null	
Driverphoto	varchar(50)	not null	
Driveremail	varchar(50)	not null	
Ambulanceregisternumber	varchar(50)	not null	

*5. location table*

Column name	Data type	Constraints	Description
Location_id	int	primary key	location identifier
Driver_id	int	foreign key	
Location	varchar(50)	Not null	
Latitude	varchar(50)	Not null	
longitude	varchar(50)	Not null	

*6. emergency table*

Column name	Data type	Constraints	Description
Emergency_id	int	primary key	unique identifier
User_id	int	foreign key	
Driver_id	int	foreign key	
Date_and_time	varchar(200)	not null	
Status	varchar(200)	not null	
Latitude	varchar(200)		
Longitude	varchar(200)		



*7.notify table*

Column name	Data type	Constraints	Description
Notify_id	Int	Primary_key	Notification identifier
User_id	Int	Foreign key	
Hospital_id	Int	Foreign key	
Date	varchar(50)	not null	
Description	varchar(200)	not null	
Casualties	varchar(200)	not null	
Type	varchar(50)	not null	

*8.report table*

Column name	Data type	Constraints	Description
Report_id	int	Primary key	Report identifier
User_id	int	Foreign key	
Driver_id	int	Foreign key	
Dateandtime	varchar(50)	Not null	
status	varchar(50)	Not null	
reason	varchar(200)	Not null	

*9.hospital table*

Column name	Data type	Constraints	Description
Hospital_id	int	primary key	hospital identifier
Hospitalname	varchar(200)	not null	Name of hospital
Hospitalphoto	varchar(200)	not null	
Hospitalplace	varchar(200)	not null	
Hospitalpost	varchar(200)	not null	
Hospitalpin	Int	not null	
Hospitalcontact	varchar(200)	not null	
Hospitallatitude	varchar(200)	not null	
Hospitallongitude	varchar(200)	not null	

*10.feedback table*

Column name	Data type	Constraints	Description
Feedback_id	Int	primary key	unique identifier
User_id	Int	Foreign key	
Driver_id	Int	Foreign key	
Dateandtime	Varchar(20)	not null	
Feedback	Varchar(200)	Not null	

### **3.5. Data Flow Diagram**

A graphical representation is used to describe and analyses the movement of data through a system manual or automated including the processes, Storing of data and delays in the system. Data flow diagrams are the central tool and the basis from which other components are developed.

The transformation of data, from input to output through process may be described logically and independently of the physical components associated with the system.

They are termed logical dataflow diagrams, showing the actual implementations and the movement of data between people, departments and

workstations. DFD is one of the most important modelling tools used in system design. DFD shows the flow of data through different process in the system.

**PURPOSE:**

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have reasonably completed model of the behavior of the system. It is important to avoid premature designs, wherein develop designs before analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

Throughout my project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In my opinion, "efficient design of the data flow and context flow diagram helps to design the system successfully without much major flaws within the scheduled time". This is the most complicated part in a project. In the designing process, my project took more than the activities in the software lifecycle. If we design a system efficiently with all the future enhancements the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as a way of expressing system requirements in graphical form. A data flow diagram also known as "bubble chare" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It functionality decomposes the requirement specification down to the lowest level. Data Flow Diagram depicts the

information flow, the transformation flow and the transformations that are applied as data move from input to output. Thus DFD describes what data flows rather than how they are processed.

Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is one of the most important tools used during system analysis. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

Data Flow Diagrams are made up of a number of symbols, which represents system components. Data flow modelling method uses four kinds of symbols, which are used to represent four kinds of system components.

These are

- Process
- Data stores
- Data flows
- External entity

**Process:**

Process shows the work of the system. Each process has one or more data inputs and produce one or more data outputs. Processes are represented by rounded rectangles in Data Flow Diagram. Each process has a unique name and number. This name and number appears inside the rectangle that represents the process in a Data Flow Diagram.

**Data Stores:**

A data stores is a repository of data. Processes can enter data, into a store or retrieve the data from the data store. Each data has a unique name.

**Data Flows:**

Data flows show the passage of data in the system and are represented by lines joining system components. An arrow indicates the direction of flow and the line is labelled by name of the dataflow.

### **External Entity:**

External entities are outside the system but they either supply input data into the system or use other systems output. They are entities on which the designer has control. They may be an organizations customer or other bodies with which the system interacts. External entities that supply data into the system are sometimes called source. External entities that use the system data are sometimes called sinks. These are represented by rectangles in the

### **Data Flow Diagram.**

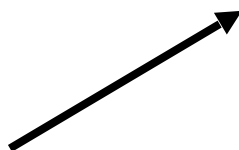
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Basic data flow diagram symbols are.....

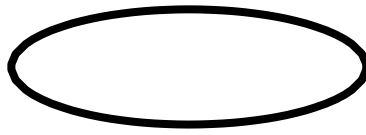
- A Square defines a source (originator) or destination of a system data:



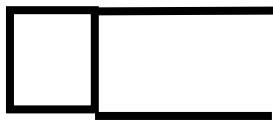
- An Arrow identifies data flow. It is a pipeline through which information flows:



- A Circle represents a process that transforms incoming data flow(s) into outgoing data flow(s):



- An Open Rectangle is a data store:



**Four steps are commonly used to construct a DFD:**

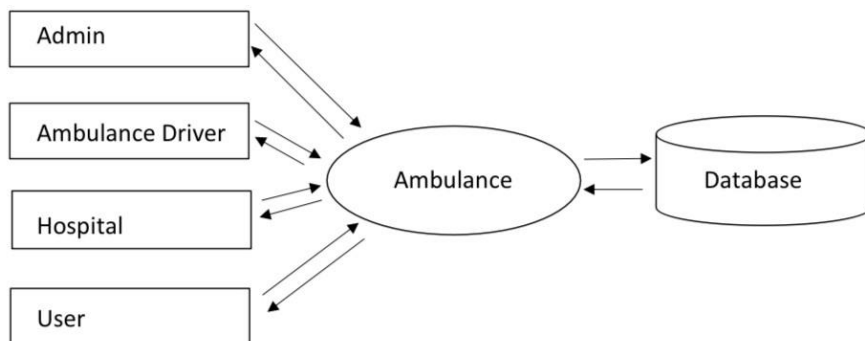
- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and left to right.
- When a process is exploded in to lower level details they are numbered.
- The names of data stores, sources and destinations are written in Capital letters.

## DFD Level-0

### Level – 0

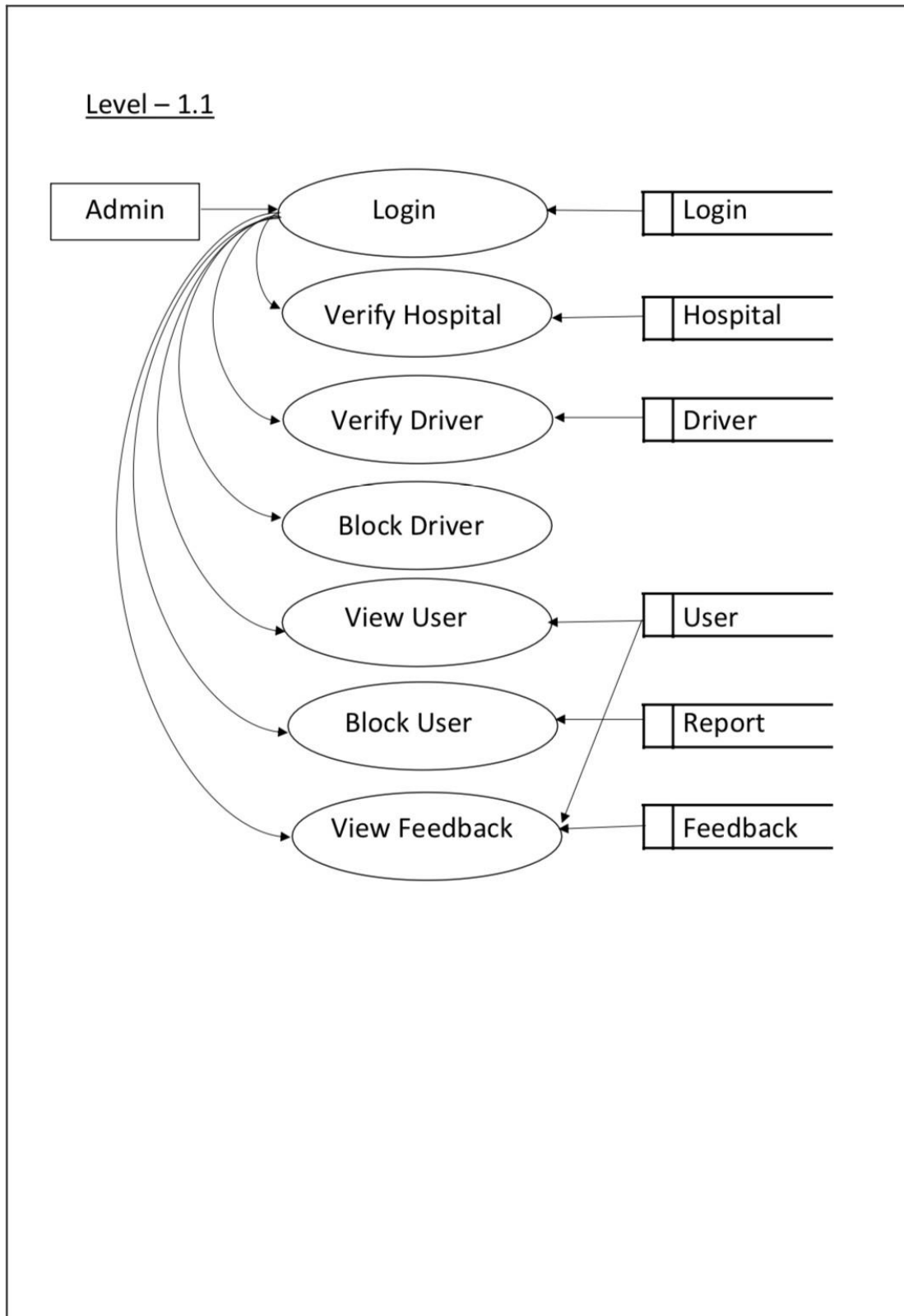


### Level – 1

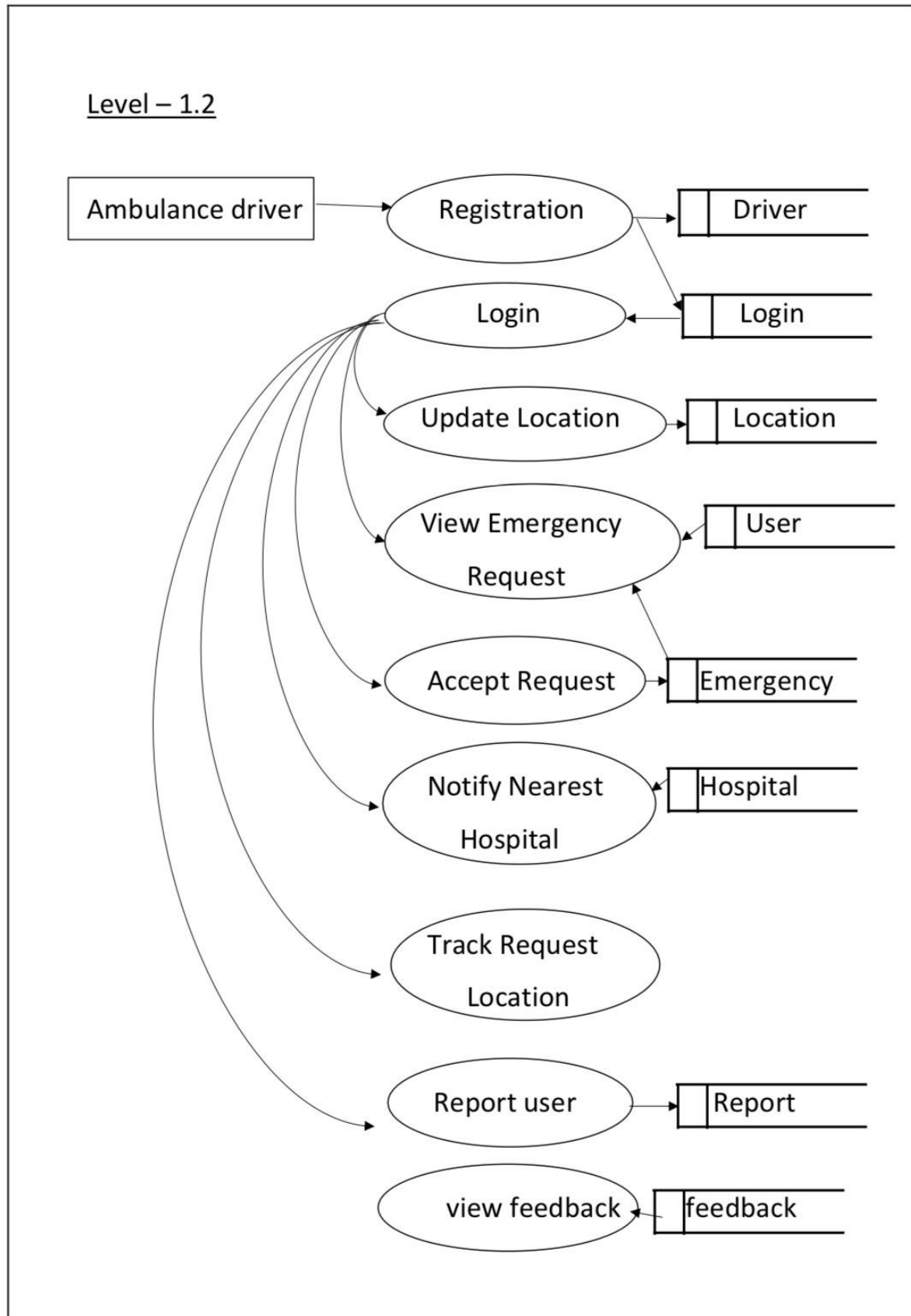




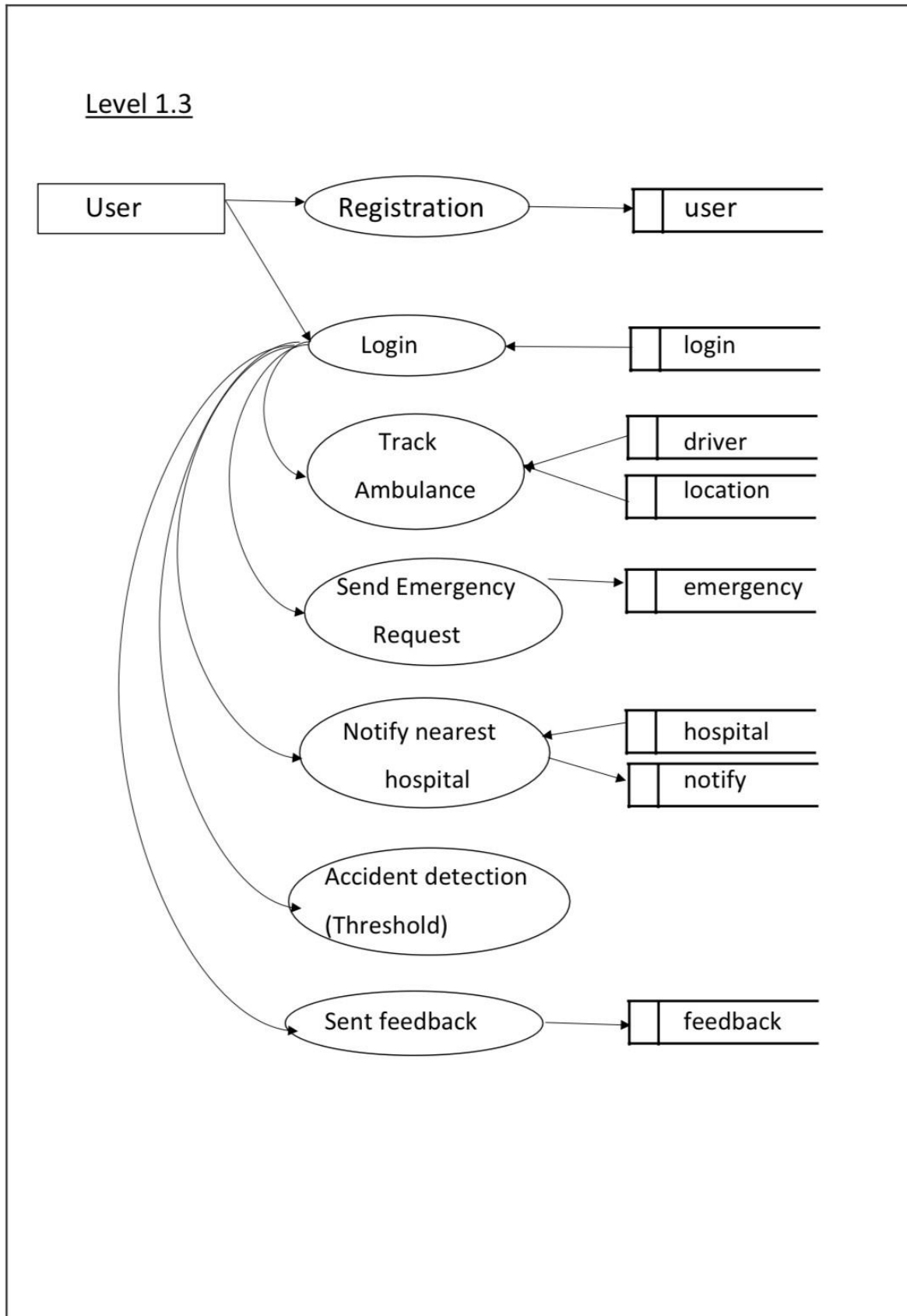
## DFD Level-1.1 Admin



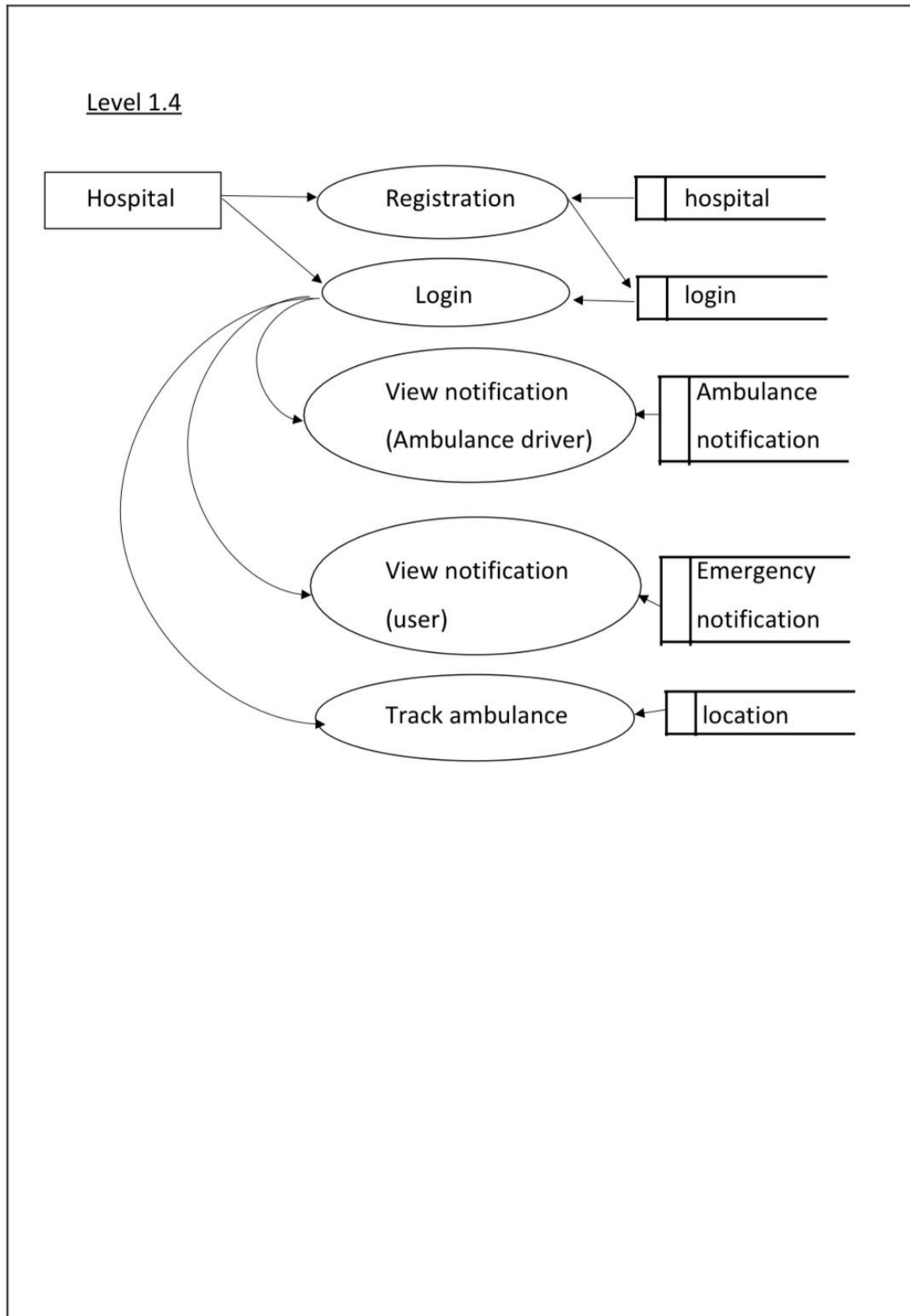
## DFD Level-1.2 Ambulance Driver



### DFD Level-1.3 USER



### DFD Level-1.3 Hospital



### 3.6 ER Diagram

An ER diagram is a diagram that helps to design databases in an efficient way. It is a data model for describing the data or information. It is a visual representation of data that describes how data is related to each other. The main components of ER models are entities, attributes and the relationships that can exist among them.

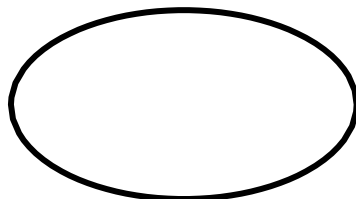
#### Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.



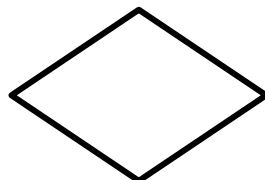
#### Attribute

Attributes are properties of entities. Attributes are represented by means of eclipses. Every eclipse represents one attribute and is directly connected to its entity (rectangle).

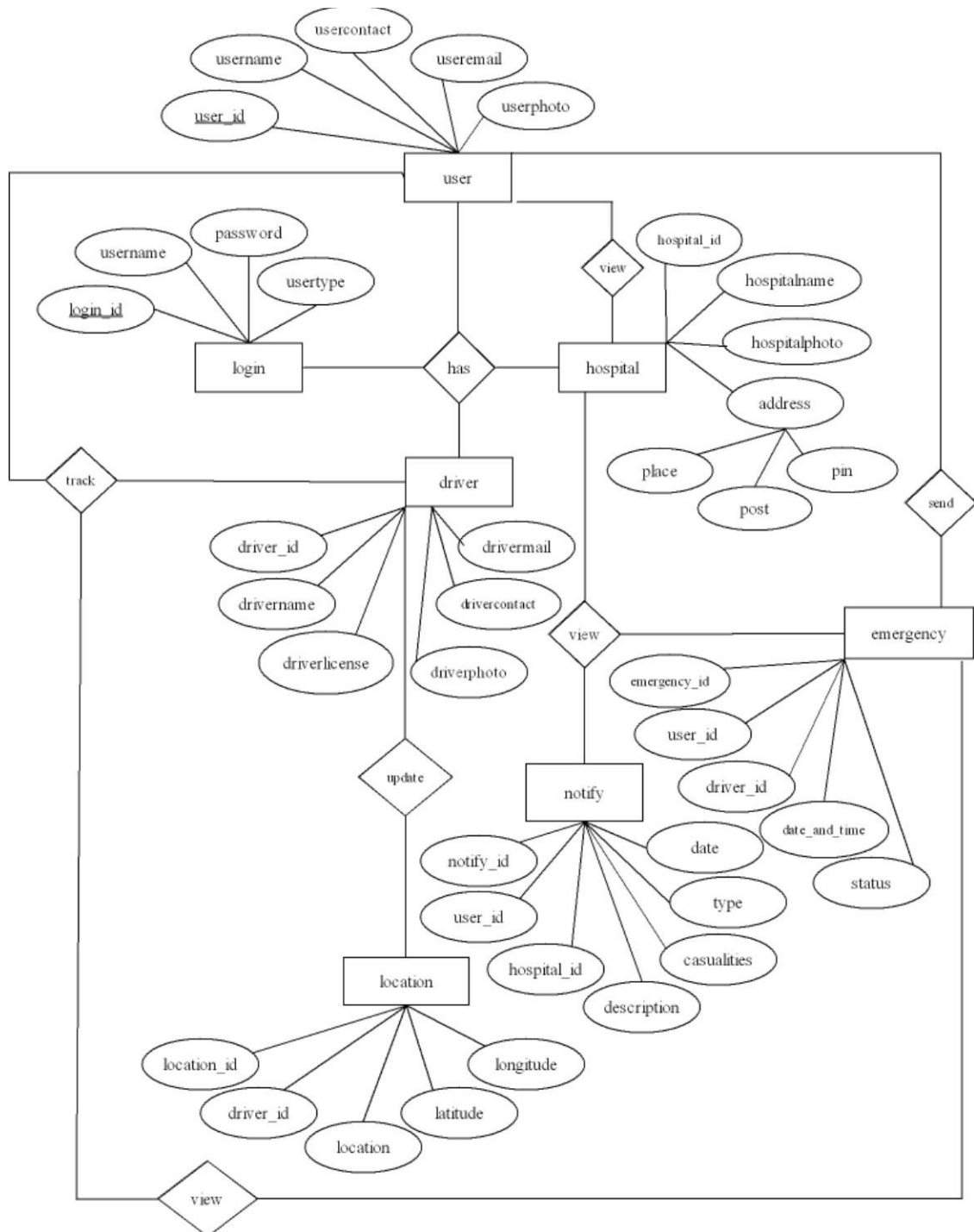


#### Relationship

Relationships are represented by diamond shaped box. Name of the relationship is written in the diamond box. All entities (rectangles), participating in relationship, are connected to it by a line.



## E-R DIAGRAM



## **4.CODING**

## **4.1 INPUT INTERFACE**

Input design is a part of overall system design, which requires very careful attention. If data going into the system is correct, then the processing and output will magnify these errors. Thus the designer has a number of clear objectives in the different stages of input design.

- To produce a cost effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that input is acceptable to and understand by the user.

## **4.2 OUTPUT INTERFACE**

At the beginning of the output design various types of outputs such as external, internal, operational and interactive and turn around are defined. Then the format, content, location, frequency, volume and sequence of the outputs are specified. The content of the output must be defined in detail. The system analysis has two specific objectives at this stage.

- To interpret and communicate the results of the computer part of a system to the users in a form, which they can understand, and which meets their requirements.
- To communicate the output design specifications to programmers in a way in which it is unambiguous, comprehensive and capable of being translated into a programming language.

## **4.3 SOFTWARE DESCRIPTION**

### **4.3.1 HTML**

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML



documents from a web server or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML

specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

HTML files are written in ASCII text, so the user can use any text editor to create his/her web page, though a browser of one sort or another is necessary to view the web page. HTML is case insensitive with its language commands. The characters within the document, however, are case sensitive. The language consists of various "tags" which are known as elements. These allow the browser to understand (and put into the desired/specified format) the layout, background, headings, titles, lists, text and/or graphics on the page. The elements are classified according to their function in the HTML document. There are head elements and body elements. The head elements identify properties of the entire document, while body elements actually mark text as content and show a change in the appearance in one way or another. Most elements have a beginning and an ending which encompass the text the user wishes to mark with the tag. All HTML documents must begin with the element and end with the element. Some of the other elements which may be used are tags to create lists--both ordered lists as well as unordered lists. The user may also create larger or smaller, bolder, italicized, or underlined text. Attributes may be used along with the elements. These perform functions such as placement of text, indication of the source files of images, and identification of links to the document or part of the document.

## **FEATURES**

- 1) It is a very easy and simple language. It can be easily understood and modified.
- 2) It is very easy to make an effective presentation with HTML because it has a lot of formatting tags.
- 3) It is a markup language, so it provides a flexible way to design web pages along with the text.

## HTML VERSIONS

Since the time HTML was invented there are lots of HTML versions in market, the brief introduction about the HTML version is given below:

**HTML 1.0:** The first version of HTML was 1.0, which was the barebones version of HTML language, and it was released in 1991.

**HTML 2.0:** This was the next version which was released in 1995, and it was standard language version for website design. HTML 2.0 was able to support extra features such as form-based file upload, form elements such as text box, option button, etc.

**HTML 3.2:** HTML 3.2 version was published by W3C in early 1997. This version was capable of creating tables and providing support for extra options for form elements. It can also support a web page with complex mathematical equations. It became an official standard for any browser till January 1997. Today it is practically supported by most of the browsers.

**HTML 4.01:** HTML 4.01 version was released on December 1999, and it is a very stable version of HTML language. This version is the current official standard, and it provides added support for stylesheets (CSS) and scripting ability for various multimedia elements.

**HTML5 :** HTML5 is the newest version of Hyper Text Markup language. The first draft of this version was announced in January 2008. There are two major organizations one is W3C (World Wide Web Consortium), and another one is WHATWG (Web Hypertext Application Technology Working Group) which are involved in the development of HTML 5 version, and still, it is under development.

## Description of HTML example

**<!DOCTYPE>:** It defines the document type or it instruct the browser about the version of HTML

**<html> :** This tag informs the browser that it is an HTML document. Text between html tag describes the web document. It is a container for all other elements of HTML except<!DOCTYPE>

**<head>** : It should be the first element inside the element, which contains the metadata (information about the document). It must be closed before the body tag opens.

**<title>**: As its name suggests it is used to add title of that html page which appears at the top of the browser window. It must be placed inside the head tag and should close immediately (optional).

**<body>** : Text between body tag describes the body content of the page that is visible to the end user. This tag contains the main content of the html document

**<h1>** : Text between <h1> tag describes the first level heading of the webpage.

**<p>**: Text between <p> tag describes the paragraph of the webpage.

### 4.3.2 CSS

#### History of CSS

CSS was first proposed by **Hakon Wium Lie** on October 10, 1994. At the time, Lie was working with Tim Berners-Lee (father of Html) at CERN. The European Organization for Nuclear Research is known as CERN. Hakonwium lie is known as father of css. CSS was proposed in 1994 as a web styling language, to solve some of the problems of Html 4. There were other styling languages proposed at this time, such as Style Sheets for Html and JSSS but CSS won.

#### Why to learn CSS?

**Cascading Style Sheets**, fondly referred to as **CSS**, is a simple design language intended to simplify the process of making web pages presentable.<sup>23</sup> CSS is a MUST for students and working professionals to become a great Software Engineer specially when they are working

in Web Development Domain.

I will list down some of the key advantages of learning CSS:

- **Become a web designer** - If you want to start a carrier as a professional web designer, HTML and CSS designing is a must skill.
- **Control web** - CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.
- **Learn other languages** - Once you understand the basic of HTML and CSS then other related technologies like javascript, php, or angular are become easier to understand.

## Types of CSS

There are three ways of inserting a style sheet in any Html documents, they are given below:

- Inline style sheet
- Internal style sheet
- External style sheet

Inline CSS is use with any elements of HTML where it is used on page. Here we use inline CSS for paragraph, the example shows how to change the color and the left margin of a paragraph. An internal style sheet should be used when a single document has a unique style. Internal styles sheet is defined in the head section of an HTML page, by using the <style> tag. An external style sheet is ideal when the style is applied to many pages. With an external style sheet, we can change the look of an entire Web site by changing one file.

## CSS Selectors

Selectors are used for select an Html element it is select by name, id, class etc.

1. id selector
2. class selector
3. Element Selector
4. Group Selector

## 5. Universal Selector

### Features

1. A style rule consists of a selector component and a declaration block component.
2. The selector is used to point to the HTML component which you want to get styled.
3. Inside the declaration block, one or more declarations are contained along with semicolons.
4. Every declaration which is put has a CSS property name, a semicolon, and a value. For example, color is the property and the value is red in color. Font size is the property and the 15px is the value.
5. CSS declaration ends with a semicolon and these blocks are surrounded by curly braces.
6. CSS selectors are the ones which are used to find HTML elements which are based on the element name, id, attribute, class and more.
7. One unique element will be selected by the ID of an element.
8. If you wish to select the particular element with a specific id, the # function along with the id attribute should be used.
9. If you wish to select the elements with a specific class, the period character along with the name class should be written.
10. Universal selector: If you are not interested in choosing the elements of a certain type, the universal selector simply matches with the element name.
11. Element selector: These selectors choose the element based on the element name.
12. Descendent selector: When a particular element lies inside another element, then it is called as the descendent selector.
13. ID selector: This selector uses the id of the HTML element so that a specific element could be selected.
14. Class selectors: It selects the element with a specific class attribute.
15. Grouping selectors: It will be a good option to group the selectors so as to minimize the code. Each selector along with a comma should be used to group the selectors.

### 4.3.3 JavaScript

**JavaScript** is a object-based scripting language and it is light weighted. It is first implemented by Netscape (with help from Sun Microsystems). JavaScript was created by Brendan Eich at Netscape in 1995 for the purpose of allowing code in web-pages (performing logical operation on client side).It is not compiled but translated. JavaScript Translator is responsible to translate the JavaScript code which is embedded in browser.Netscape first introduced a JavaScript interpreter in Navigator 2.

#### Why we use JavaScript?

Using HTML we can only design a web page but you cannot run any logic on web browser like addition of two numbers, check any condition, looping statements (for, while), decision making statement (if-else) at client side. All these are not possible using HTML so for perform all these task at client side you need to use JavaScript **History** JavaScript is an object-based scripting language and it is light weighted. It is first implemented by Netscape (with help from Sun Microsystems). JavaScript was created by Brendan Eich at Netscape in 1995 for the purpose of allowing code in web-pages (performing logical operation on client side).Using HTML we can only design a web page but you cannot run any logic on web browser like addition of two numbers, check any condition, looping statements(for, while), decision making statement(if-else) etc. All these are not possible using HTML so to perform all these task, we use JavaScript. Using HTML we can only design a web page if we want to run any programs like c programming we use JavaScript. Suppose we want to print sum of two numbers then we use JavaScript for coding.

#### Features

- JavaScript is an object-based scripting language.
- Giving the user more control over the browser.
- It Handling dates and time.
- It Detecting the user's browser and OS
- It is light weighted.
- JavaScript is a scripting language and it is not java.

- JavaScript is case sensitive.
- JavaScript is object based language as it provides predefined objects.
- Most of the JavaScript control statements syntax is same as syntax of control statements in C language. An important part of JavaScript is the ability to create new functions within scripts. Declare a function in JavaScript using function keyword
- Rich Interface to your site visitors.

#### 4.3.4 MySQL

MySQL is an open-source relational database management system (RDBMS). MySQL is released under an open-source license. So you have nothing to pay to use it. MySQL is a very powerful program in its own right.

It handles a large subset of the functionality of the most expensive and powerful database packages. It uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.

It works very quickly and works well even with large data sets. It is very friendly to PHP, the most appreciated language for web development. It supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB). It is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

##### **Major features as available in MySQL 5.6**

- A broad subset of ANSI SQL 99, as well as extensions.
- Cross-platform support.
- Stored procedures, using a procedural language that closely adheres to SQL/PSM.
- Triggers.
- Cursors.
- Updatable views.
- Online DDL when using the InnoDB Storage Engine.
- Information schema.
- Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
- A set of SQL Mode options to control runtime behaviour, including a strict mode to better adhere to SQL standards.



- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine.
- Transactions with savepoints when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
- ACID compliance when using InnoDB and NDB Cluster Storage Engines.
- SSL support → Query caching → Sub-SELECTs (i.e. nested SELECTs) .
- Built-in replication support (i.e., master-master replication and master-slave replication) with one master per slave, many slaves per master.
- Multi-master replication is provided in MySQL Cluster, and multimaster support can be added to unclustered configurations using Galera Cluster.
- Full-text indexing and searching.
- Embedded database library.
- Unicode support.
- Partitioned tables with pruning of partitions in optimizer.
- Shared-nothing clustering through MySQL Cluster.
- Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.
- Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.
- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

## **Advantages**

MySQL database server has lots of advantages over its competitors. Some of these advantages have been explained below.

➤ Open Source and Cost Effective:

The best thing about MySQL server is that this is open source and it has a free version as well.

By open source software, we mean that the code of the software is available and anyone can tailor it according to his requirement. Companies prefer MySQL because they don't have to pay anything for this excellent product.

➤ Portability:

MySQL is cross platform database server. MySQL can be run on a variety of platforms including Windows, OS2, Linux and Solaris. Portability of MySQL server makes it suitable for applications that target multiple platforms particularly web application. MySQL contains API for almost all the major programming languages and can be easily integrated with the languages like PHP, C++, Perl, C, Python and ruby. In fact, MySQL is a part of the famous LAMP (Linux Apache MySQL PHP) server stack which is used worldwide for web application development.

➤ Seamless Connectivity:

Various secure and seamless connection mechanisms are available in order to connect with MySQL server. These connections include named pipes, TCP/IP sockets and UNIX Sockets.

➤ Rapid Development and Continuous Updates:

Being an open source product, MySQL has a very large developer community which releases regular patches and updates for MySQL. Several database templates have been developed which can be readily used and modified resulting in rapid application development.

➤ Security:

MySQL server databases are extremely secure and all the data access scenarios are protected via password and good thing about these passwords is that they are stored in encrypted form and it is not easy to break these advanced and complex encryption algorithms.

### 4.3.5 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

#### Major features of python

- Easy to code
- Free and Open Source
- Object-Oriented Language
- GUI Programming Support
- High-Level Language
- Extensible feature
- Python is **Portable** language

- Python is Integrated language

### **Advantages**

- Extensive support libraries
- Integration feature
- Improved productivity
- Easy to learn and write
- Vast library support
- Free and open source

### **4.3.6 Flask**

Flask is a web application framework written in Python. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Pocco. Flask is based on the Werkzeug WSGI toolkit and the Jinja2 template engine. Both are Pocco projects.

It is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier. It gives developers flexibility and is a more accessible framework for new developers since you can build a web application quickly using only a single Python file. Flask is also extensible and doesn't force a particular directory structure or require complicated boilerplate code before getting started.

## **5.CODING PAGES**

## 5.1 Admin page

```
from flask import Flask, render_template, redirect, request, session, jsonify
from DBConnection import Db
import random
import datetime

app = Flask(__name__)
app.secret_key="hii"
# @app.route('/')
# def abc():
#     return render_template("index.html")

@app.route('/', methods=['get'])
def login():
    return render_template("index.html")

@app.route('/login1', methods=['post'])
def logint1():
    u=request.form['textfield']
    p=request.form['textfield2']
    db=Db()
    res=db.selectOne("select * from login where username='"+u+"' and
password='"+p+"'")
    if res is not None:
        if res["usertype"]=="admin":
            session['lg']="lin"

            return redirect('/admin_home')
        elif res["usertype"]=="hospital":
            session['lid']=res['login_id']
            return redirect('/hospital_home')

    else:
        return "<script>alert('Unauthorised User');window.location='/';</script>"

    else:
        return "<script>alert('Invalid details');window.location='/';</script>"

@app.route('/logout')
def logout():
    session.clear()
    session['lg']=" "
    return redirect('/')

@app.route('/admin_home')
def admin_home():
    return render_template('Admin/index.html')
```

```

@app.route('/view_approved_driver')
def view_approved_driver():
    if session['lg']=="lin":
        db = Db()
        qry = db.select("select * from driver,login where driver.Driver_id=login.login_id
and (login.usertype='driver' or login.usertype='blocked')")
        return render_template('Admin/view_approved_driver.html',data=qry)
    else:
        return redirect('/')

```

```

@app.route("/block_driver/<did>")
def block_driver(did):
    if session['lg']=="lin":
        db=Db()
        db.update("update login set usertype='blocked' where login_id='"+did+"'")
        return redirect("/view_approved_driver")
    else:
        return redirect('/')

```

```

@app.route("/unblock_driver/<did>")
def unblock_driver(did):
    if session['lg'] == "lin":
        db=Db()
        db.update("update login set usertype='driver' where login_id='"+did+"'")
        return redirect("/view_approved_driver")
    else:
        return redirect('/')

```

```

@app.route('/view_driver')
def view_driver():
    if session['lg'] == "lin":
        db = Db()
        qry = db.select("select * from driver,login where driver.Driver_id=login.login_id
and login.usertype='pending'")
        return render_template('Admin/view_driver.html',data=qry)
    else:
        return redirect('/')

```

```

@app.route("/approve_driver/<did>")
def approve_driver(did):
    if session['lg'] == "lin":
        db=Db()
        db.update("update login set usertype='driver' where login_id='"+did+"'")
        return redirect("/view_driver")
    else:
        return redirect('/')

```

```

@app.route("/reject_driver/<did>")
def reject_driver(did):
    if session['lg'] == "lin":
        db=Db()
        db.delete("delete from login where login_id='"+did+"'")
        db.delete("delete from driver where Driver_id='"+did+"'")

```

```

        return redirect("/view_driver")
    else:
        return redirect('/')

@app.route('/view_feedback')
def view_feedback():
    if session['lg'] == "lin":
        db = Db()
        qry=db.select("select * from feedback,user,driver where
feedback.User_id=user.User_id and feedback.Driver_id=driver.Driver_id")
        return render_template('Admin/view_feedback.html',data=qry)
    else:
        return redirect('/')

@app.route('/view_hospital')
def view_hospital():
    if session['lg'] == "lin":
        db = Db()
        qry=db.select("select * from hospital,login where
hospital.Hospital_id=login.login_id and login.usertype='pending'")
        return render_template('Admin/view_hospital.html',data=qry)
    else:
        return redirect('/')

@app.route("/reject_hospital/<hid>")
def reject_hospital(hid):
    if session['lg'] == "lin":
        db=Db()
        db.delete("delete from login where login_id='"+hid+"'")
        db.delete("delete from hospital where Hospital_id='"+hid+"'")
        return redirect("/view_hospital")
    else:
        return redirect('/')

@app.route("/approve_hospital/<hid>")
def approve_hospital(hid):
    if session['lg'] == "lin":
        db=Db()
        db.update("update login set usertype='hospital' where login_id='"+hid+"'")
        return redirect("/view_hospital")
    else:
        return redirect('/')

@app.route('/view_approved_hospital')
def view_approved_hospital():
    if session['lg'] == "lin":
        db = Db()
        qry=db.select("select * from hospital,login where
hospital.Hospital_id=login.login_id and (login.usertype='Hospital' or
login.usertype='blocked'")
        return render_template('Admin/view_approved_hospital.html',data=qry)
    else:

```



```

    return redirect('/')

@app.route("/block_hospital/<hid>")
def block_hospital(hid):
    if session['lg'] == "lin":
        db=Db()
        db.update("update login set usertype='blocked' where login_id='"+hid+"'")
        return redirect("/view_approved_hospital")
    else:
        return redirect('/')

@app.route("/unblock_hospital/<hid>")
def unblock_hospital(hid):
    if session['lg'] == "lin":
        db=Db()
        db.update("update login set usertype='hospital' where login_id='"+hid+"'")
        return redirect("/view_approved_hospital")

@app.route('/view_user')
def view_user():
    if session['lg'] == "lin":
        db = Db()
        qry=db.select("select * from user,login where user.User_id=login.login_id and login.usertype='user'")
        return render_template('Admin/view_user.html',data=qry)
    else:
        return redirect('/')

@app.route("/admin_view_report")
def admin_view_report():
    if session['lg'] == "lin":
        db=Db()
        res=db.select("select report.*, user.Username, driver.Drivername, login.usertype from report, user, driver, login where report.User_id=user.User_id and report.Driver_id=driver.Driver_id and user.User_id=login.login_id order by report.Report_id DESC")
        return render_template("Admin/view_report.html", data=res)
    else:
        return redirect('/')

@app.route("/block_user/<uid>")
def block_user(uid):
    if session['lg'] == "lin":
        db=Db()
        db.update("update login set usertype='block' where login_id='"+uid+"'")
        return redirect("/admin_view_report")
    else:
        return redirect('/')

@app.route("/unblock_user/<uid>")
def unblock_user(uid):

```

```

if session['lg'] == "lin":
    db=Db()
    db.update("update login set usertype='user' where login_id='"+uid+"'")
    return redirect("/admin_view_report")
else:
    return redirect("/")

```

## 5.2 Hospital page

```

@app.route('/reg',methods=['get'])
def reg():
    return render_template("registration.html")

@app.route('/reg_post',methods=['post'])
def reg_post():
    name=request.form['textfield']
    image=request.files['fileField']
    place=request.form['textfield2']
    post=request.form['textfield3']
    pin=request.form['textfield4']
    contact=request.form['textfield5']
    email=request.form['textfield6']
    latitude=request.form['textfield7']
    longitude=request.form['textfield8']
    pwd=random.randint(0000,9999)
    date=datetime.datetime.now().strftime("y%m%d-%H%M%S")
    image.save(r"C:\Users\anude\PycharmProjects\G-Vaahan\static\pic\\"+date+'.jpg')
    path="/static/pic/"+date+'.jpg'
    db=Db()
    res=db.insert("insert into login values ('','"+email+"','"+str(pwd)+"','hospital')")

    db.insert("insert into hospital values
('"+str(res)+"','"+name+"','"+str(path)+"','"+place+"','"+post+"','"+pin+"','"+contact+
','"+email+"','"+latitude+"','"+longitude+"')")
    return "ok"

@app.route('/view_noti')
def view_noti():
    db=Db()
    res=db.select("select notify.*, driver.Drivername,
driver.Drivercontact,location.Latitude,location.longitude from notify, driver,location
where driver.Driver_id=notify.User_id and notify.Type='driver' and
notify.Hospital_id='"+str(session['lid'])+"' and location.driver_id=driver.driver_id")
    return render_template("hospital/view notification(ambulance).html",data=res)

@app.route('/view_noti_user')
def view_noti_user():
    db=Db()
    res = db.select("select notify.*, user.Username, user.Usercontact from notify, user
where user.User_id=notify.User_id and notify.Type='user' and notify.Hospital_id='"+
str(session['lid']) + "'")
    return render_template("hospital/view notification(user).html",data=res)

```

```
@app.route('/hospital_home')
def hospital_home():
    return render_template("hospital/index.html")
```

## **6.SYSTEM TESTING**

## 6.1 TESTING AND EVALUATION

Testing is a process of executing a program with the intent of finding an error. Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design and coding. Testing includes verifications of the basic logic of each program and verification that the entire system works properly. Testing demonstrates that software functions appear to be working according to specification. In addition, data collected as testing is conducted provides a good indication of software quality as a whole. The debugging process is the most unpredictable part of testing process. Testing begins at the module level and works towards the integration of the entire computer-based system. Testing and debugging are different activities, but any testing includes debugging strategy for software testing must accommodate low level tests that are necessary to verify that a small source code segment has been currently implemented as well as high-level tests that validate major system function, against customer requirements. No testing is complete without verification and validation part. The goals of verification and validation activities are to access and improve the quality of work products generated during the development and modification of the software. There are two types of verification: life cycle verification and formal verification. Life cycle verification is the process of determining the degree to which the products of the given phase of the development cycle fulfil the specification established during the prior process. Formal verification is the rigorous mathematical demonstration that source code conforms to its specifications. Validation is a process of evaluating software at the end of the software development process to determine conformance with the requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The primary objectives, when we test software are the following:

- Testing is a process of executing with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.

- A successful test is one uncovers undiscovered errors.

Thus, testing plays a very critical role in determining the reliability and efficiency of the software and hence is very important stage in software development. Tests are to be conducted on the software to evaluate its performance under a number of conditions. Ideally, it should so at the level

of each module and also when all of them are integrated to form the completed system. Software testing is done at different levels.

## **6.2 TESTING STRATEGIES**

A strategy for software testing integrates software test case design method into a well-planned series of steps that result in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote a customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project progresses.

**The general characteristics of software testing strategies are:**

- Testing begins at the component level and works “outward” toward the integration of the entire computer system.
- Different testing techniques are appropriate at different points in time.

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

A strategy must provide guidance for the practitioner and set of milestones for the manager. Because the step on the test strategy occurs at a time when deadline pressure begins to rise, progress must be measurable and problem must surface as early as possible.

The software teams approach to testing is defining a plan that describes an overall strategy and a procedure that defines specific testing steps and tests that will be conducted. In the proposed system, if the administrator makes any attempt to login to the application without entering his password, then the system will not allow the user to login to the application.

## **6.3 TESTING TECHNIQUES**

The various testing techniques are given below:

### **6.3.1 WHITE BOX TESTING**

White-box testing is also called as glass-box testing, is a test case design method that goes to the control structure of the procedural design to derive

test cases. Using white box testing methods, the software engineer can derive test cases that,

- ✓ Guarantee that all independent paths within a module have been exercised at Least once.
- ✓ Exercise all logical decision on their true and false sides.
- ✓ Execute all loops at their boundaries and within their operational sides.
- ✓ Exercise internal data structure to ensure their validity.

White box testing was successfully conducted on our system. All independent paths within a module have been executed at least once and all logical decisions have been exercised on their true and false sides.

### **6.3.2 BLACK BOX TESTING**

Black-box testing is also called as behavioural testing, focuses on the functional requirement of the software. It is a complimentary approach that is likely uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories.

- ✓ Incorrect or missing functions.
- ✓ Interface errors.
- ✓ Error on data structures or external database access.
- ✓ Behaviour or performance errors.
- ✓ Initialization and termination errors.

Black box testing was successfully conducted on your system. The system was divided into a number of modules and testing was conducted on each module. We have tested the system for incorrect or missing functions, interface and performance errors.

### **6.3.3 UNIT TESTING**

Unit testing comprises the set of tests performed by an individual programmer prior to the integration of the system. Testing removes residual bugs and improves the reliability of the system.

Testing allows the developer to find out the design faults if any, and enable correction if needed. Exhaustive unit testing has to be carried out to ensure the validity of the data. In order to successfully test the entire package, unit testing is carried out. Each module was tested as when it was developed. Thus it proved easier to conduct minute testing operation and correct them then and there.



### **6.3.4 INTEGRATION TESTING**

Bottom-up integration is the traditional strategy used to integrate the component of a software system into a functional whole. Bottom-up integration consists of unit testing, followed by subsystem testing and followed by testing of the entire system. Unit testing has the goal of discovering errors in the individual parts of the system.

Parts are tested in isolation from one another in an artificial environment known as “Test Harness”, which consists of driver programs and data necessary to exercise the modules. Unit testing should be as exhaustive as possible to ensure that each representative case handled by each module has been tested. Unit testing is eased by a system structure that is composed of small loosely coupled modules.

Both control and data interfaces must be tested. Large software system may require several levels of subsystem testing. Lower level subsystems are successively combined to form higher level subsystems. In most software systems, exhaustive testing of subsystem capabilities is not feasible due to the combination complexity of the module interfaces. Therefore, test cases must be carefully chosen to exercise the interfaces in the desired manner.

### **6.3.5 ACCEPTANCE TESTING**

Acceptance testing involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements. It is not unusual for two sets of acceptance test to be run, those developed by the quality group and those developed by the customer.

In addition to the functional and performance tests, stress tests are performed to determine the limitation of the system. For example, a compiler might be tested to determine the effect of the symbol table overflow, or real-time system might be tested to determine the effect of simultaneous arrival of numerous high priority interrupts.

## **OUTPUT TESTING**

Output testing of the proposed system is important since no system could be useful if it does not produce the required output.

The output format on the screen is found to be correct, as the format was designed in the system design phase according to the user needs. For the hard copy also the output comes out as the specified requirements by the user. Hence output testing doesn't result in any correction on the system.

## **7.SYSTEM IMPLEMENTATION AND DEPLOYMENT**

Implementation is the process of deploying the new system in the operational environment. Proper implementation is essential to provide a reliable system to meet the organizational requirement. There are four types of implementation methods. They are Direct Changeover, Phased Implementation, Parallel Run and Pilot Approach. The most commonly used implementation methods are Pilot Approach and Parallel Run.

The system which is developed as a web application hence the other functions as normal application, as usual some web development technologies are used in the implementation of the project. The language I selected to program this software is PHP. The reason I selected PHP is that is a simple and powerful language that especially developed to create web application.

Technologies used in the development of the software are:

- ✓ Programming language – Python
- ✓ DBMS – MySQL server
- ✓ Development tool – Py charm
- ✓ Development platform – Windows 11

The front end is HTML, and CSS and back end is MySQL Server and Python. The system developed on Pycharm in Windows 10 operating system.

## **8.CONCLUSION**

The “G-VAAHAN” has been developed for all given conditions and it is found working effectively under the all the circumstances that may arise in the real environment. The software has been developed to reducing the operator work.

This system is user friendly and is well efficient to make easy interaction with the users of system. The system is done with an insight into the necessary modifications that may be required in the future. Hence the system can be maintained successfully without much work.

## **8.1 FUTURE ENHANCEMENT**

As a future venture, it is suggested to make some changes to provide more services and to provide information at right time in right manner.

The current system is designed to connect ambulance drivers and hospitals in a locality, but we will upgrade this to a wide area that will be useful for many people. We will ensure that more hospitals and ambulance drivers connected to this project named G-Vaahan. We will also make the automatic accident detection more accurate by new methods

## **9.REFERENCE**

### **Online Reference**

- Google
- Youtube

### **Books Reffered**

- Python programming:A beginner's guide to learn python from zero
- Learn android development with android studio

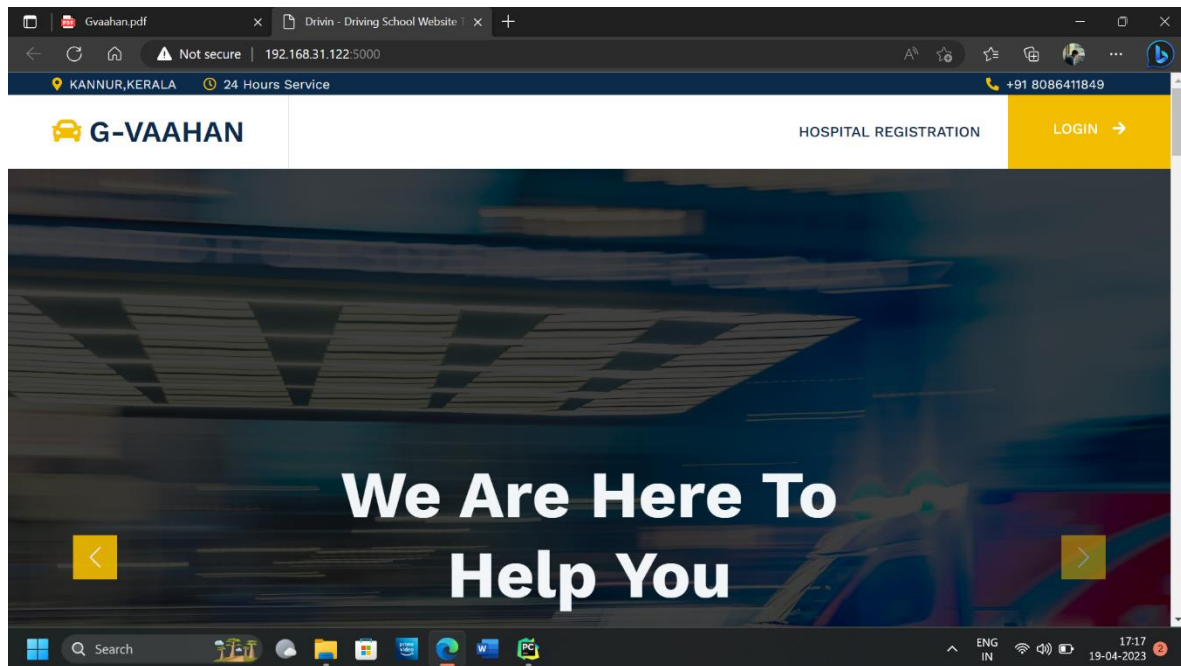
### **Websites**

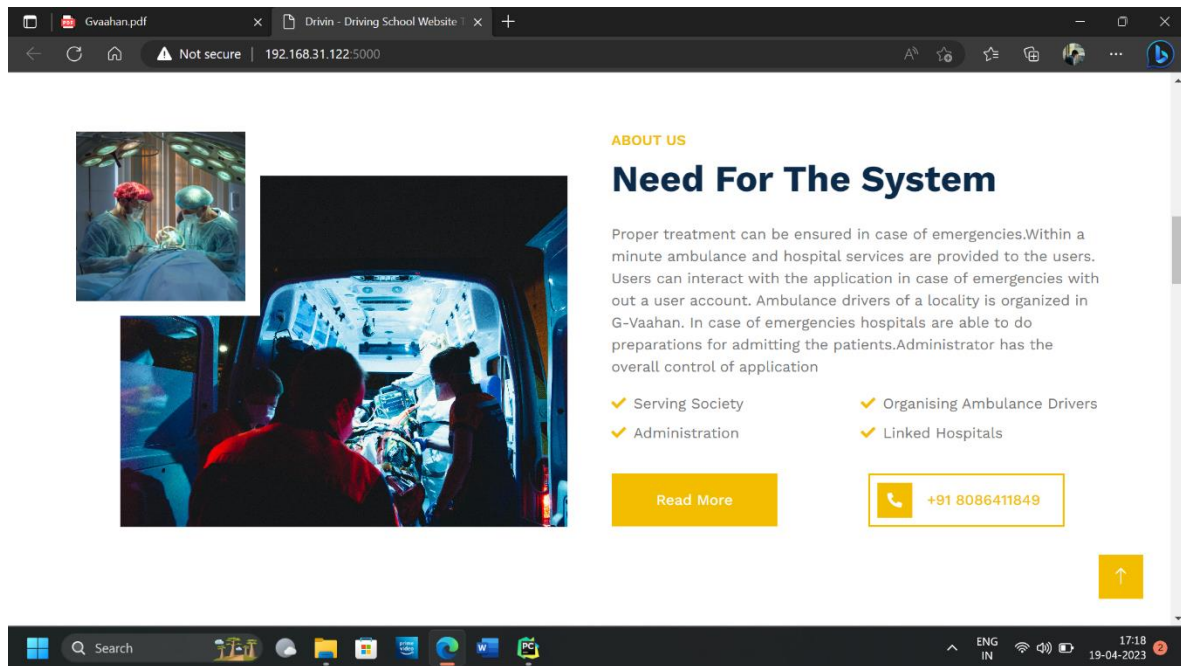
- W3Schools.com
- Python.org
- Stackoverflow.com

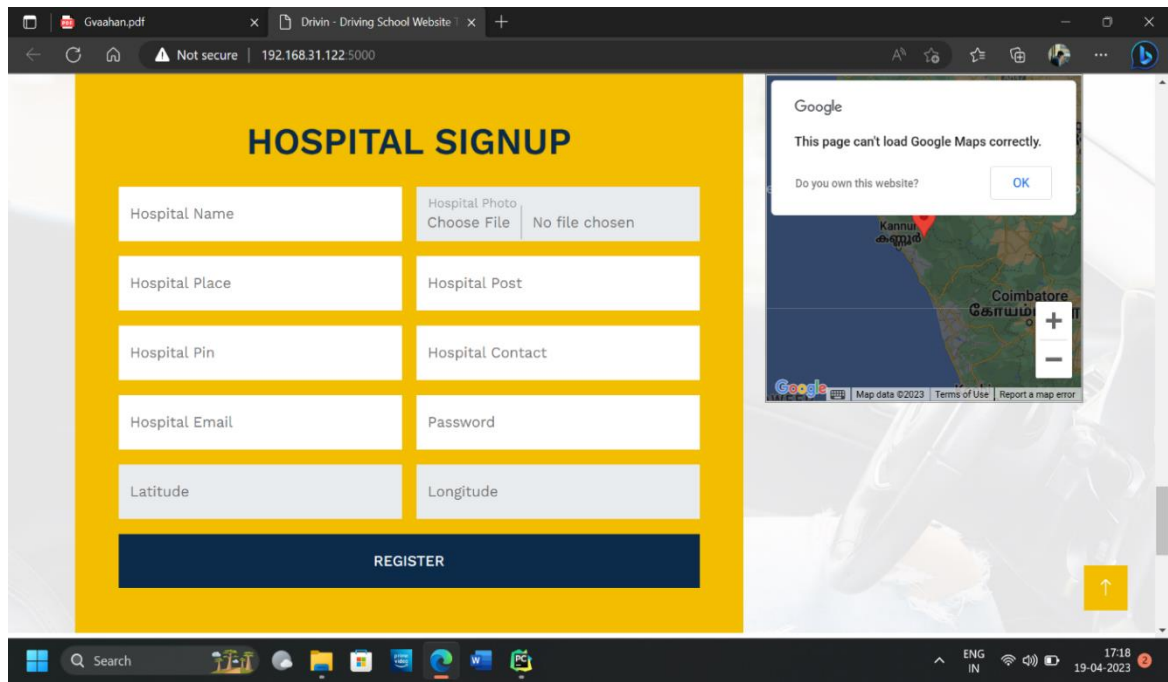


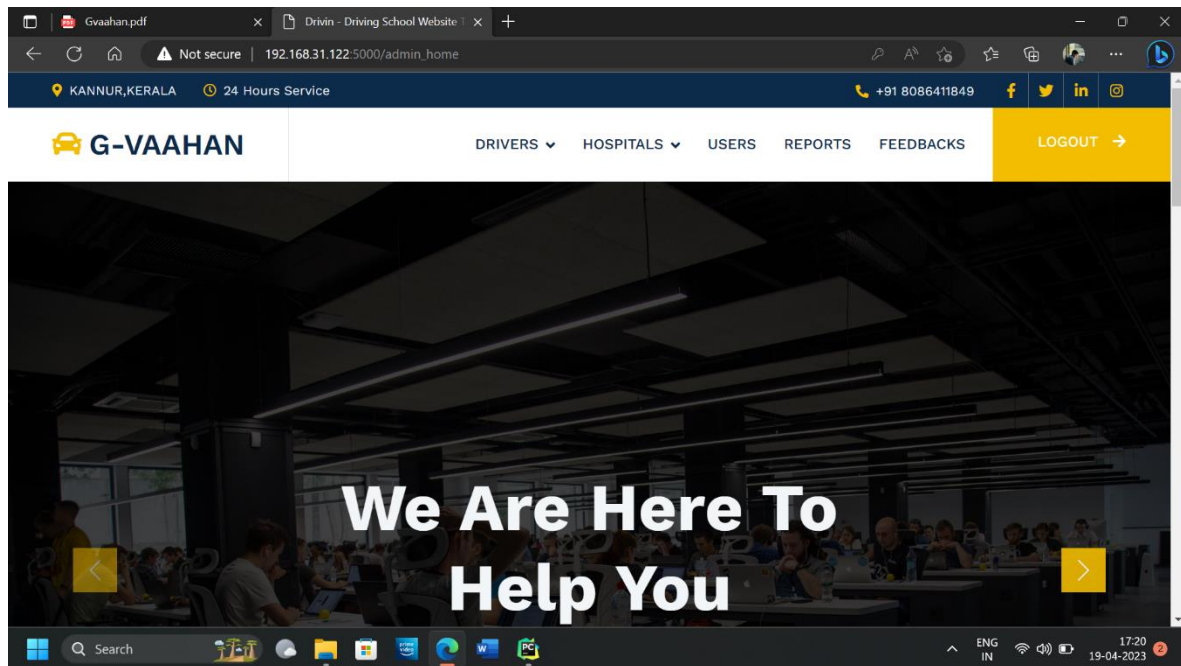
## **10.APPENDIX**

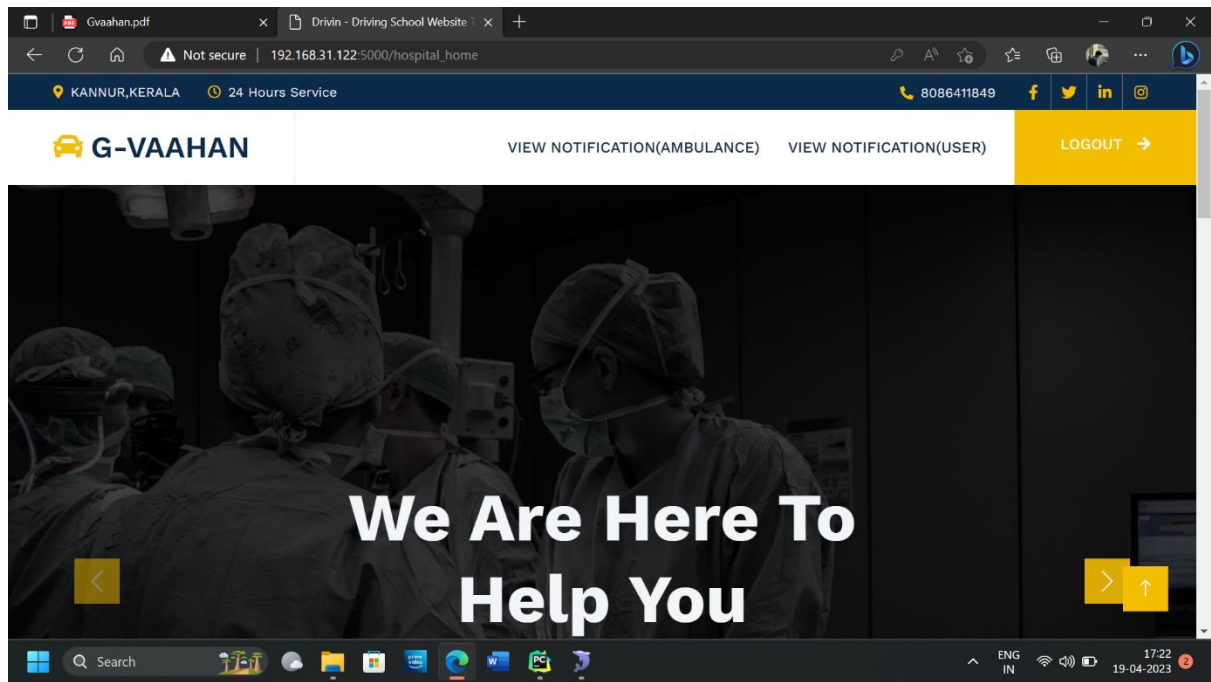
### 10.1.1 Homepage:

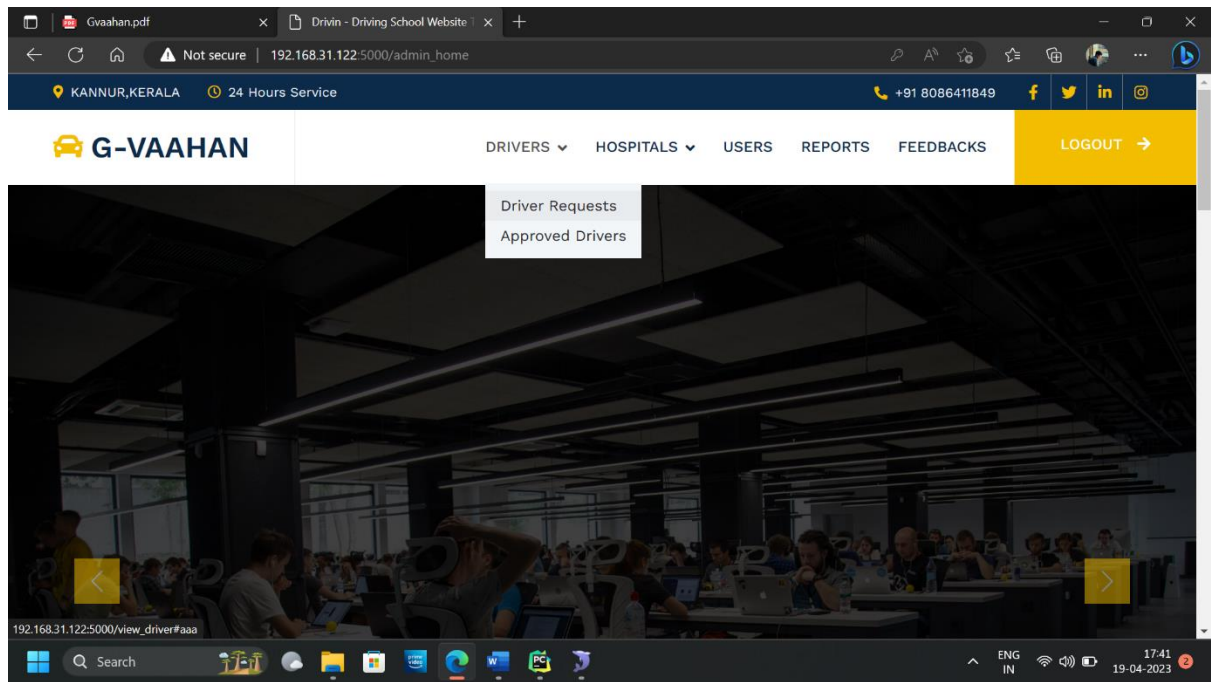


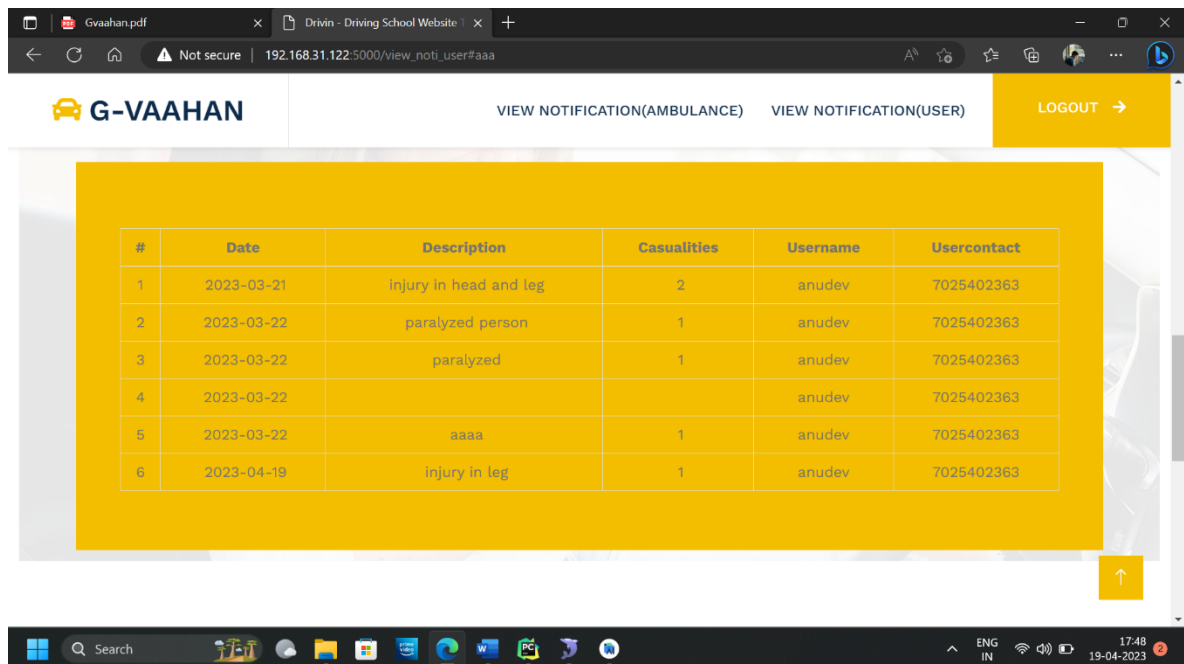
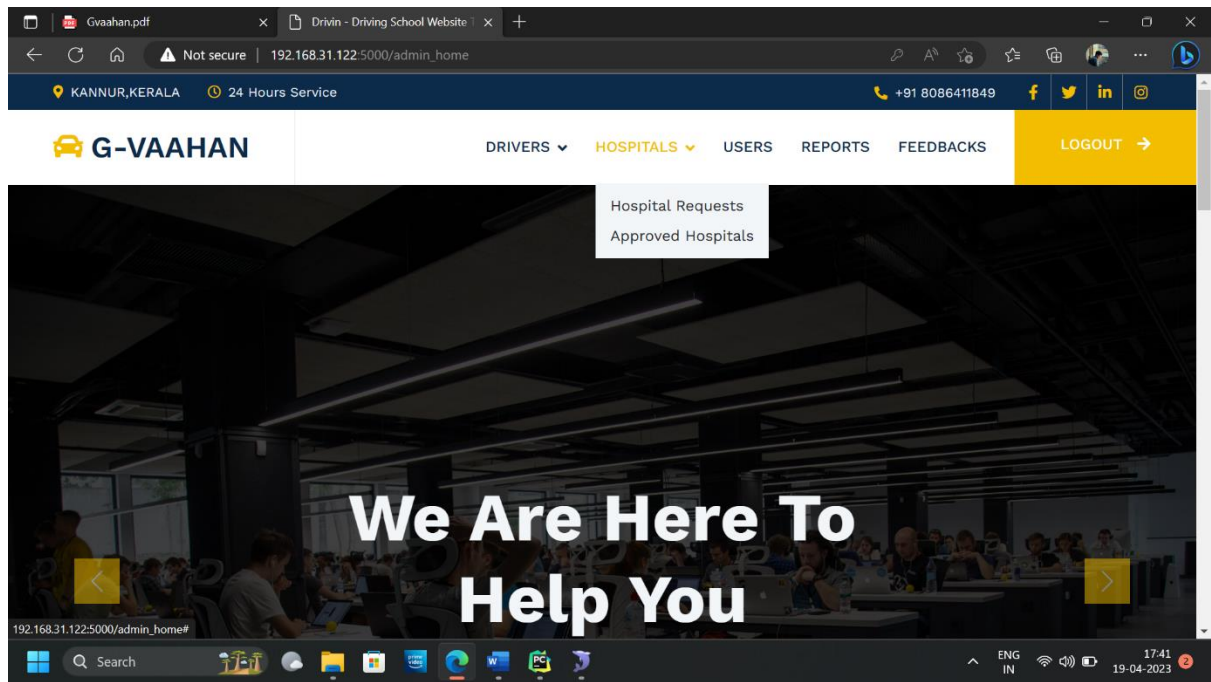
















G-VAAHAN

DRIVERS ▾ HOSPITALS ▾ USERS REPORTS FEEDBACKS

LOGOUT →

#	User Name	Photo	Contact	Email
1	anudev		7025402363	Anudev@gmail.com
2	bixon		2147483647	bixx@gmail.com

↑

G-VAAHAN

VIEW NOTIFICATION(AMBULANCE) VIEW NOTIFICATION(USER)

LOGOUT →

#	Date	Description	Casualties	Drivername	Drivercontact	
1	2023-03-21	severe injury in leg and stomach	1	deon	+91 7034049567	<a href="#">Track</a>

↑

Gvaahan.pdf × Drivin - Driving School Website 1 × +

Not secure | 192.168.31.122:5000/admin\_view\_report#aaa

**G-VAAHAN** DRIVERS ▾ HOSPITALS ▾ USERS REPORTS FEEDBACKS **LOGOUT** →

#	Date	Driver Name	Report Against	Reason	
1	2023-03-22	deon	anudev	wrong call	<b>Block</b>
2	2023-02-27	cv	anudev	fake emergency request	<b>Block</b>
3	2022-12-06	deon	anudev	Fake emergency message sent	<b>Block</b>


↑

Search ENG IN 17:42 19-04-2023

Gvaaahan.pdf

Drivin - Driving School Website

Not secure | 192.168.31.122:5000/view\_feedback#aaa




DRIVERS ▾HOSPITALS ▾USERSREPORTSFEEDBACKS

LOGOUT →

#	User Name	Driver Name	Date	Feedback
1	anudev	deon	2023-03-18	good driver

↑

Search



ENG  
IN

17:42  
19-04-2023



**A PROJECT REPORT ON**  
**GROOT**  
**FURNITURE APP BASED ON AUGMENTED**  
**REALITY**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ANUPAM R**

**REG.NO: DB20BCAR23**

**MACKEY ABDUL RAHEEM TP**

**REG.NO: DB20BCAR10**

**ASHWINI JITHESH**

**REG.NO: DB20BCAR14**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**

**A PROJECT REPORT ON**  
**GROOT**  
**FURNITURE APP BASED ON AUGMENTED**  
**REALITY**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ANUPAM R**

**REG.NO: DB20BCAR23**

**MACKEY ABDUL RAHEEM TP**

**REG.NO: DB20BCAR10**

**ASHWINI JITHESH**

**REG.NO: DB20BCAR14**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**

**DON BOSCO ARTS & SCIENCE COLLEGE**  
**ANGADIKADAVU**  
**IRITTY, KANNUR**



**CERTIFICATE**

*Certified that this report titled **GROOT-FURNITURE APP BASED ON AUGMENTED REALITY** is a bonafide record of the project work done by **Anupam R (Reg.No:DB20BCAR23)**, **Mackey Abdul Raheem TP (Reg.No:DB20BCAR10)** and **Ashwini Jithesh (Reg.No:DB20BCAR14)** under the supervision and guidance, towards partial fulfilment of the requirement for award of the degree of bachelor of computer application (BCA) of the Kannur university.*

Project Guide

Head of the Department

Angadikadavu

External Examiner

Date:

- 1.
- 2.

## Declaration

We, ANUPAM R, MACKEY ABDUL RAHEEM TP and ASHWINI JITHESH, sixth semester BCA student of Don Bosco Arts & Science College, Angadikadavu, under Kannur University do hereby declare that the project entitled **“GROOT-FURNITURE APP BASED ON AUGMENTED REALITY”** is the original work carried out by me in the sixth semester under the supervision of Mr. Hebin Layola, Lecturer of the Department of BCA, Don Bosco Arts & Science College, Angadikadavu, in partial fulfilment of the requirement for the award of the degree Bachelor of Computer Application, Kannur University.

Angadikadavu

Date

ANUPAM R

MACKEY ABDUL RAHEEM TP

ASHWINI JITHESH



# ACKNOWLEDGEMENT

First of all we thank the lord almighty for his immense grace and blessings showered on me at every stages of this work. I am greatly indebted to our Principal Fr. Dr. Francis Karackat SDB, Don Bosco Arts & Science College, Angadikadavu for providing the opportunity to take up this project as part of my curriculum.

We deeply indebted to my project guide Ms. Sruthi Nimesh, lectures of department of BCA, for her assistance and valuable suggestions as guide. She made this project a reality.

We express our sincere thanks to Mrs. Sindu PM, Mr. Hebin Layola, Mrs. Fincy Cyriac and Mrs. Vineetha Mathew, lecturers of department of BCA, for their valuable suggestions during the course of this project. Their critical suggestions helped me to improve the project work.

Acknowledging the efforts of everyone, their chivalrous help in the course of the project preparation and their willingness to collaborate with the work, their magnanimity through lucid technical details lead to the successful completion of my project.

We would like to express my sincere thanks to all my friends, colleagues, parents and all those who have directly or indirectly assisted during this work.

# CONTENTS

chapters	contents	Page No
1	Introduction	9
2	System Analysis	16
2.1	Existing System	17
2.1.1	Disadvantage Of Existing System	17
2.2	Proposed System	18
2.2.1	Advantage Of Proposed System	18
2.3	Feasibility Study	18
2.3.1	Economic Feasibility	19
2.3.2	Technical Feasibility	19
2.3.3	Behavioural Feasibility	20
2.4	System Specification	20
2.4.1	Software Specification	20
2.4.2	Hardware Specification	21
2.5	Identification Of Actors	21
2.6	Identification Of Use Cases	22
2.6.1	Use Cases For The Actor Shop	22
2.6.2	Use Cases For The Actor user	23
2.6.3	Use Cases For The Actor Staff	24
2.6.4	Use Case Diagram	25

3	SYSTEM DESIGN	27
3.1	Introduction	28
3.2	Database Design	28
3.3	Table Design	29
3.4	Groot	30
3.5	Data Flow Diagram	33
3.6	ER Diagram	41
4	CODING	43
4.1	Input Interface	44
4.2	Output Interface	44
4.3	Software Description	44
4.3.1	HTML	44
4.3.2	CSS	45
4.3.3	JavaScript	46
4.3.4	MySQL	47
4.3.5	Python	50
4.3.6	Flask	51
5	CODING PAGES	52
5.1	Shop Page	53
5.2	Staff Page	60
6	System Testing	62
6.1	Testing And Evaluation	63
6.2	Testing Strategies	64

6.3	Testing Techniques	65
6.3.1	White Box Testing	65
6.3.2	Black Box Testing	65
6.3.3	Unit Testing	66
6.3.4	Integration Testing	66
6.3.5	Acceptance Testing	66
6.3.6	Output Testing	67
7	SYSTEM IMPLEMENTATION AND DEPLOYMENT	68
8	CONCLUSION	69
9	REFERENCE	70
10	APPENDIX	71

# **1.INTRODUCTION**

## **1.1 Project Overview**

In this fastest moving world, preserving time is one of the tiring process in day to day life. We depend on online shopping sites in order to preserve the time which might be wasted while going directly to the shops. Using this Furniture App we tries to implement a fully online shopping site based on Augmented Reality. Augmented Reality gets our attention mostly due to its “cool factor.” Now it’s becoming more of a reality, and forward-thinking retail brands are incorporating Augmented Reality technology into the customer experience. The Application is meant for a single furniture shop who can manage furniture. And also this application provides customers with a better shopping experience so that he/she can choose the most suitable furniture for their home without even going to the shops but simply by checking it using their smart phones with the help of augmented reality. Furniture App is a completely an online application so that all transactions could be done under user-convenience.

## **NEED FOR THE SYSTEM**

One of the main benefits of integrating AR into your ecommerce is that your customers are experiencing a much more interactive shopping experience - making them more likely to return and recommend your site to friends and family. Not only are they getting a feel for the product before seeing it in real life with the new ‘try before you buy’ outlook, but customers are able to interact with the product in a new and exciting way... whilst also being able to customise the product to fit their own sense of style! Not only can a customer visualise a product, it’s size, colour and proportions compared to their own living space - but shoppers will have a significantly calmer online shopping experience as AR removes the anxiety customers have surrounding buying furniture online. AR and 3D Product Customisers are completely transforming the sales cycle, and how large items are bought and sold.

## **OBJECTIVES AND SCOPE**

The application aims to achieve a number of objectives. The project objectives of this application are as follow:

### **1. Enhance customer's online shopping experience**

The project aims to provide users with real-time live view of the furniture wherever they are using their smartphones. Hence, after the user selects a product from the list, the user can then activate the smartphone's camera to view the selected product in AR. Therefore, users can immediately recognise how well is the product and whether it meets their needs and preferences, without the hassle of travelling out or going to showrooms.

### **2. Provide realistic 3D view of the featured product**

It is not enough if users can only view the furniture from one angle, thus, the project should be able to calibrate the 3D model with the information given to place the furniture correctly. Users can also then move around the furniture to get a clearer picture on how it would look like from multiple perspectives. Furthermore, the application should anchor the object to a specific position on the plane surface where the camera is directed to. This will avoid the furniture to move together with the smartphone, making it difficult to view from another angle.

### **3. Provide accurate live size scale at 1:1 ratio to real-world environment**

The project must be able to show the furniture at live size so that users can ensure the furniture fits into the physical space available. In addition, bringing furniture into users' household helps to create experiential learning of the augmented furniture, bringing up users' desires. Therefore, providing a live size view helps to create a realistic experience for users as if they have acquired the product, but disappears immediately when closing the application.

### **4. Allow easy maintenance of the application**

As the project is used on online platform, managers must be able to perform maintenance on the application easily. Thus, managers should be able to add models into the application's database, which should then reflect immediately on the next data retrieval on the application. Hence, users will be able to view new products instantly and take a good look on the product using AR.

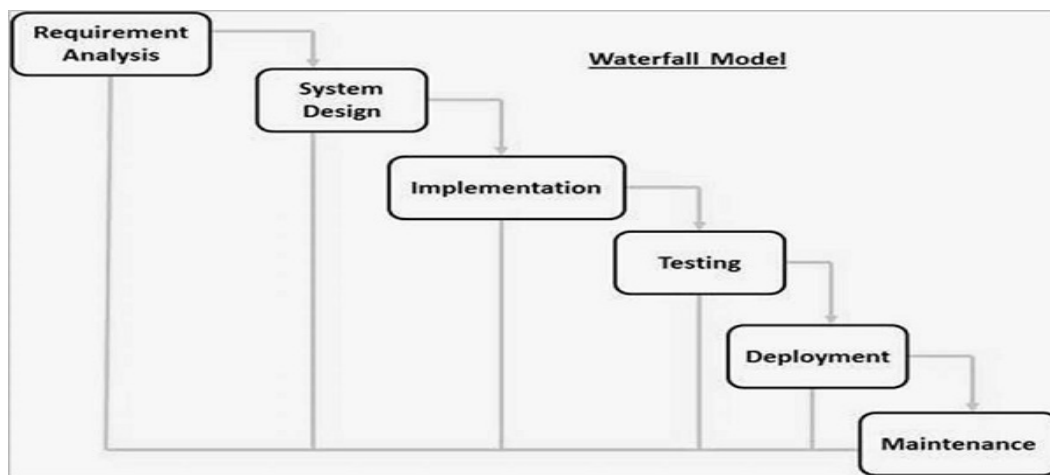
## **MODEL:**

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software

development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially

### **Waterfall Model - Design:**

In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. Following is the pictorial representation of Iterative and Incremental model:



The sequential phases in Waterfall model are –

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying



hardware and system requirements and helps in defining the overall system architecture.

- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

### **Waterfall Model - Application:**

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are –

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.

- The project is short.

**The advantages of the waterfall model SDLC Model are as follows:**

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Some of the major advantages of the Waterfall Model are as follows –

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

**The disadvantages of the waterfall model SDLC Model are as follows:**

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows –

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

## **2. SYSTEM ANALYSIS**

## **Introduction**

System analysis is the process of collecting and interpreting facts, understanding problems and using the information to suggest improvement on the system. This will help to understand the existing system and determine how computers make its operation more effective. The aim of this analysis is to collect detailed information on the system and the feasibility study of the proposed system.

### **2.1 EXISTING SYSTEM**

The existing furniture shopping system is very full of effort the customers have to go to the market and then check different shops in order to buy what they want. Sometimes they have to try even ten shops and yet unable to buy the same furniture of the required design or color suited with their house.

The Existing online furniture shopping allow the users to see only the different images of the product and these data is useless to select a furniture. Research has found that many of the customers choose furniture in a hasty fashion and it cause buying the wrong sized item for their room

#### **2.1.1 The Disadvantages of Existing System**

The existing system has the following disadvantages,

- Consumers have to travel a certain distance for the product in the case of traditional shopping
- High transportation charge
- Purchase through online there will be higher chance to buy the wrong sized item for our room
- Customers can't see the product physically instead of that they can see only the different images of the product

## **2.2 PROPOSED SYSTEM**

A web based application where user, have to place the marker in a room where they want to try out furniture items. The user's webcam will be on and through the webcam they will capture the live feed of the room. Application captures the image and passes through predefined marker detection algorithm. Algorithm is based on image processing techniques using color and other properties as the input to detect the marker. User initially selects the furniture to be placed from the given database. The application superimposes furniture on the original image with the center coinciding with the markers center in both directions. Furniture objects are overlaid on to the two dimensional image frame acquire from webcam. This will appear as if it is actually placed in the real world. And finally the user can view how the area looks with the furniture present.

### **2.2.1 Advantages of Proposed System**

- Find and try products remotely
- Explore brand new ways to shop
- Engage and retain customers
- Saves time
- Reduces travel expenses

## **2.3 Feasibility Study**

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioral feasibilities.

The proposed system is theoretically investigated to check the feasibility and found that they are more reliable and efficient in the cases given below. There are three aspects in the feasibility study portion of the preliminary investigation.

✓ Economic feasibility

✓ Technical feasibility

✓ Behavioural feasibility

The proposed system must be evaluated from a technical point of view first,

and if technical feasible their impact on the organization must be assessed. If compatible, the operational system can be devised. Then they must be tested for economic feasibility.

### **2.3.1 Economic Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors which affect the development of a new system is the cost it would require. Since the system developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

### **2.3.2 Technical Feasibility**

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, procedures and staff. Having identified an outline system, the investigation must go on suggest the type of equipment, required method developing the system, of running the system once it has been designed. The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Though the technology become obsolete after some period of time, due to the fact that newer version of some software supports older versions, the system still be used. So there are only minimal constraints involved with this project. The system has been developed using C# and .NET, along with the database software SQL server, thus we could conclude that the project is technically feasible for development.

### **2.3.3 Behavioural Feasibility**

People are inherently resistant to change and computers have been known to facilitate change. The System is designed in user friendly manner and we need to provide any special training for the persons using this software. The operating system used is Windows 10, which is also user friendly. Since the application is web biased and can easily accessed in a web browser, which is quite familiar to the intended users, it does not have any operational barriers. So no need to provide any special training for using this application software and hence it is behaviourally feasible.

## **2.4 System Specifications**

System Specification deals with the technical aspects the project has to meet in minimum to work successfully. This also includes the different aspects the software requirement is determined from. The technical details typically include:

- Software Specification
- Hardware Specification

### **2.4.1 Software Specifications**

The software required for the application depends on the following factors:

- ✓ The flexibility of the software
- ✓ Software contracts
- ✓ Limitation of the software

## **Software Requirement**

This specifies the minimum software requirements for implementing the system. This includes:

- Front End: - python,java,XML
- Back End: - MySQL
- Client side scripting: java script



- Server side scripting: Python
- Platform:- Flask, JetBrains PyCharm, Android Studio
- Operating System: Microsoft windows 7 or above

#### **2.4.2 Hardware Specifications**

The software required for the application depends on the following factors:

- ✓ Determining size and capacity requirements.
- ✓ Computer evaluation and measurement.
- ✓ Financial factors.
- ✓ Maintenance and support.

#### **Hardware Requirement**

- Microprocessor: Intel Pentium Core i3 and above
- Clock speed: - 2.13GHz
- Ram: 4 GB and above
- Hard disk: 320 GB and above
- Keyboard:- standard keyboard
- Mouse: Standard mouse
- Connectivity: - LAN & Wi-Fi

### **2.5 Identification of Actors**

A use cases represents the functionality of an actor. It is defined as a set of actions performed by a system, which yields an observable result. An ellipse containing its name inside the ellipse or below it represents. it. It is placed inside the system boundary and connected to an actor with an association. This shoes how the use cases and the actor interact.

We can identify the actors through a list of questions. The answers to these questions bring out the actors of the system is.

- Shop
- User

- Staff

Here we need to specify the use cases of each actor.

## **2.6 Identification of use cases**

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yield an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

To find out the use cases, ask the following questions to each of the actors.

- ✓ Which functions does the actor require from the system? What does the actor need to do?
- ✓ Does the actor need to read, create, destroy, modify or store some kind of information in the system?
- ✓ Does the actor have to calculate something? And want to provide information for others?
- ✓ Could the actor's daily work be simplified or made more efficient by adding new functions to the system (typically functions which are currently not automated in the system)?

### **2.6.1 Use cases for the actor Shop**

#### **Login:**

The first step involved is login. The admin can login to the website using username and password.

#### **Manage Product:**

Admin can add or remove product.

#### **Manage Staff:**

Admin can add or remove staff.

#### **Manage Staff salary:**

Admin can manage salary of staff.

**View bill request:**

Admin can view bill.

**View order and staff allocation:**

Admin can view order and allocate staff.

**View work status:**

Admin can view work status.

## **2.6.2 Use cases for the actor user**

**Register:**

The user can register to the app.

**Login:**

The user can login to the website using username and password.

**View profile:**

User is able to view the profile.

**View furniture:**

User can view furniture.

**Add to cart:**

User can add furniture to cart.

**View cart:**

User can view cart.

**Payment:**

User can pay for booked furniture.

**View bill:**

User can view bill.

**2.6.3 Use cases for the actor Staff****Login:**

Staff can login to the website using username and password.

**View allocated work:**

Staff can view allocated work.

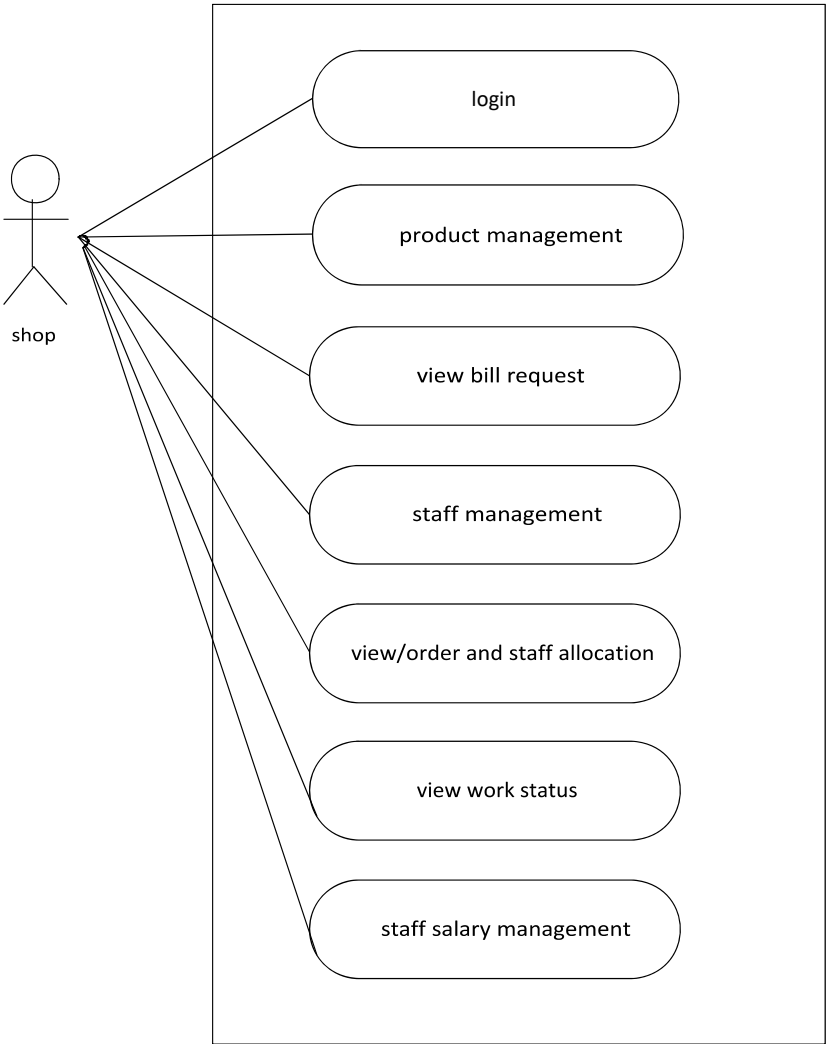
**Update work status:**

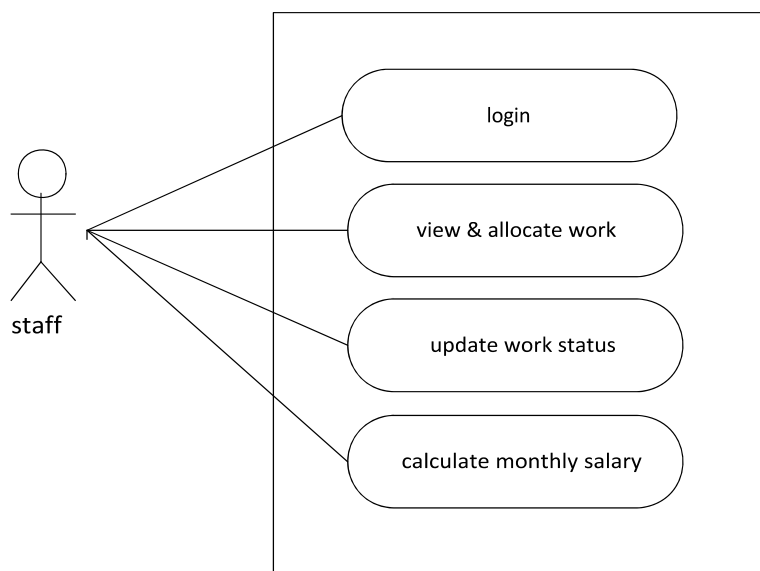
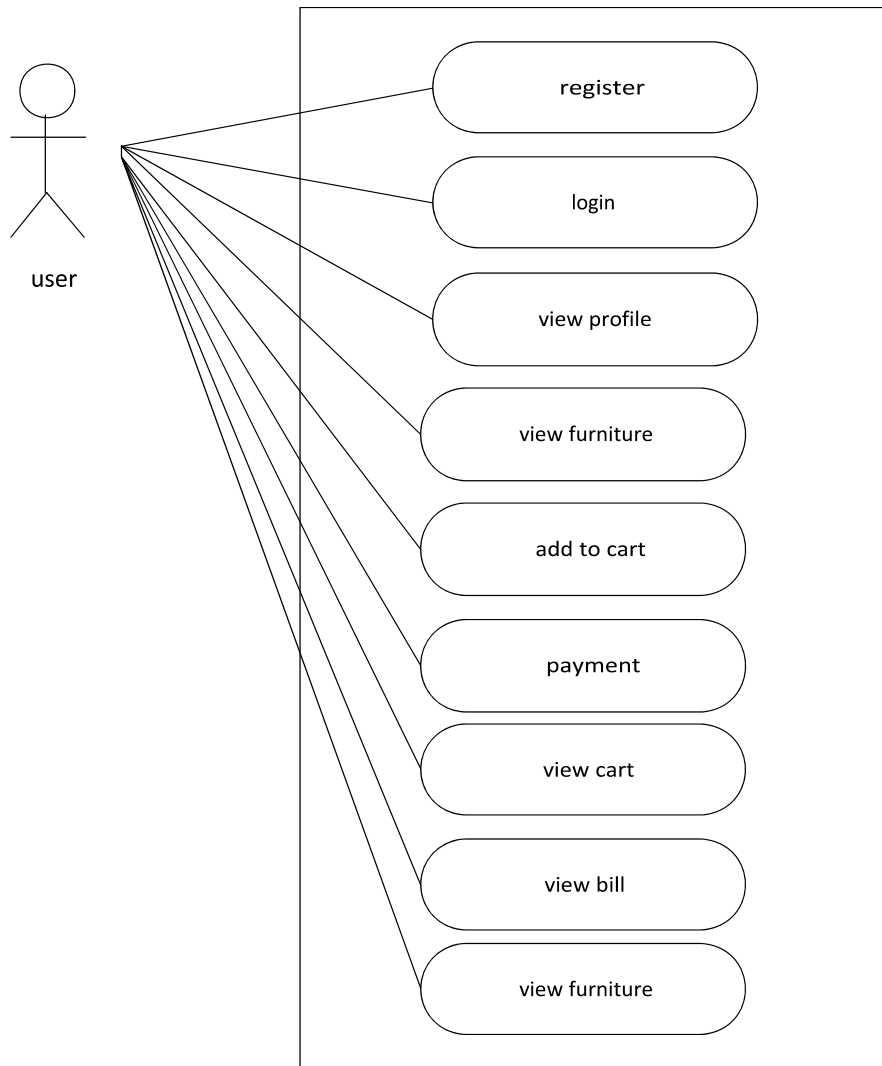
Staff can view status of work.

**Calculate monthly salary:**

Staff can calculate monthly salary

2.6.7 USE CASE DIAGRAM





### **3. SYSTEM DESIGN**

### **3.1 INTRODUCTION:**

System design provides an understanding of the procedural details, necessary implementing the system recommended in the feasibility study. Basically it is all about the creation of a new system. This is a critical phase since it decides the quality of the system and has a major impact on the testing and implementation phases.

**System design consists of three major steps.**

- Drawing of the expanded system data flow charts to identify all the processing functions required.
- The allocation of the equipment and the software to be used.
- The identification of the test requirements for the system.

### **CHARACTERS OF DESIGN**

- A design should exhibit a hierarchical organization that makes intelligent use of control among components of the software.
- A design should be modular that is, the software should be logical.
- A design should contain distinct and separable representation of data and procedure.
- A design should lead to interface that reduce the complexity of the connections between modules and with the external environment.

### **3.2 Database Design**

A Database is a collection of inter related data stored with minimum redundancy to serve many users quickly and efficiently. In database design data independence, accuracy, privacy and security are given higher priority. Database design is an integrated approach to the file design. This activity deals with the design of the physical data base. All entities and attributes have been identified while creating the database. The database design deals with the grouping of data into number of tables so as to.

- ✓ Reduplication of data.
- ✓ Minimize storage space.
- ✓ Retrieve the data efficiently.



Following are some guidelines for the database design:

- Design a relational schema so that it is easy to explain its meaning. Do not combine attributes from multiple entity and relationship type into a single relation.
- Design the database schema so that no insertion, deletion or modification anomalies are present in the relation.
- As far as possible, avoid placing attributes in the base relation whose values may frequently be null.
- Design relation schema so that they can be joined with equality conditions on attributes that are either primary keys or foreign keys in a way that no spurious tuples are generated.

### **3.3 Table Design**

DB design is required to manage large bodies of information. The management of data involves both the definition of the structure of storage of information and provisions of mechanism for the manipulation of information. For developing an efficient database certain conditions have to be fulfilled such as:

- Control Redundancy
- Ease of Use
- Data Independence
- Accuracy and Integrity

There are five major steps in design process:

- Identify the table and relationship.
- Identify the data that is needed for each table and relationship.
- Resolve the relationship.
- Verify the design.
- Implement the design

The Database Consist of the following tables given below.

### 3.4 Automated Helmet and Mask Violation

*1.Login table*

Colum name	Data type	Constraints	Description
login_id	int	primary key	unique identifier
username	varchar(50)	not null	name of user
password	varchar(50)	not null	secret key
usertype	varchar(50)	not null	To specify the role

*2. Staff table*

Colum name	Data type	Constraints	Description
staff_id	int	primary key	unique identifier
s_name	varchar(50)	not null	Name of staff
s_place	varchar(50)	not null	Place of staff
s_post	varchar(50)	not null	Post of staff
s_pin	int	not null	Pin of staff
s_phone	int	not null	Phone number of staff
s_email	varchar(50)	not null	Email of staff

*3. User table*

Colum name	Data type	Constraints	Description
user_id	int	primary key	unique identifier
user_name	varchar(50)	not null	Name of user
place	varchar(50)	not null	Place of user
house_name	varchar(50)	not null	House name of user
pin	int	not null	Pin of user
photo	varchar(50)	not null	Photo of the user
gender	varchar(50)	not null	Gender of user
email	varchar(50)	not null	Email of user

phone	int	not null	Phone number of user
-------	-----	----------	----------------------

#### *4.Product table*

Colum name	Data type	Constraints	Description
product_id	int	primary key	unique identifier
product_name	varchar(50)		Name of product
price	int		Price of product
image	varchar(50)	not null	Photo of product
details	varchar(50)	not null	Details about product
stock	varchar(50)		Number of stock of product

#### *5. Salary table*

Colum name	Data type	Constraints	Description
salary_id	int	primary key	unique identifier
staff_id	int	foreign key	
amount	int	not null	
date	date	not null	

#### *6. Cart table*

Colum name	Data type	Constraints	Description
cart_id	int	primary key	unique identifier
product_id	int	foreign key	Id of product
user_id	int	foreign key	Id of user
quantity	int	not null	Quantity of product
date	date	not null	Date
status	varchar(50)	not null	Status of product

*7. Book table*

Colum name	Data type	Constraints	Description
book_id	int	primary key	unique identifier
cart_id	int	foreign key	Id of cart
date	date	not null	Date
status	varchar(50)	not null	Status of product
quantity	int	not null	Quantity of product

*8. Bill table*

Colum name	Data type	Constraints	Description
bill_id	int	primary key	unique identifier
cart_id	int	foreign key	Id of cart
date	varchar(50)	not null	Date
amount	int	not null	Amount payed

*9. Payment table*

Colum name	Data type	Constraints	Description
payment_id	Int	primary key	unique identifier
bill_id	Int	foreign key	Id of bill
date	Date	not null	Date

*10. Allocation table*

Colum name	Data type	Constraints	Description
allocation_id	int	Primary key	unique identifier
cart_id	int	Foreign key	Id of cart
staff_id	int	Foreign key	Id of staff
status	varchar(50)	Not null	Status of ordered product
date	date	Not null	Date

### *11. Bank table*

Colum name	Data type	Constraints	Description
bank_id	int	Primary key	unique identifier
bank_name	varchar(50)	Not null	Name of bank
acc_no	int	Not null	Account number
ifsc	varchar(50)	Not null	IFSC code
amount	int	Not null	Bank balance
holder_id	int	Not null	Id of holder

### **3.5. Data Flow Diagram**

A graphical representation is used to describe and analyses the movement of data through a system manual or automated including the processes, Storing of data and delays in the system. Data flow diagrams are the central tool and the basis from which other components are developed.

The transformation of data, from input to output through process may be described logically and independently of the physical components associated with the system.

They are termed logical dataflow diagrams, showing the actual implementations and the movement of data between people, departments and

workstations. DFD is one of the most important modelling tools used in system design. DFD shows the flow of data through different process in the system.

#### **PURPOSE:**

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have reasonably completed model of the behavior of the system. It is important to avoid premature designs, wherein develop designs before analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

Throughout my project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In my opinion, "efficient design of the data flow and context flow diagram helps to design the system successfully without much major flaws within the scheduled time". This is the most complicated part in a project. In the designing process, my project took more than the activities in the software lifecycle. If we design a system efficiently with all the future enhancements the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as a way of expressing system requirements in graphical form. A data flow diagram also known as "bubble chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It functionality decomposes the requirement specification down to the lowest level. Data Flow Diagram depicts the

information flow, the transformation flow and the transformations that are applied as data move from input to output. Thus DFD describes what data flows rather than how they are processed.

Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is one of the most important tools used during system analysis. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

Data Flow Diagrams are made up of a number of symbols, which represents system components. Data flow modelling method uses four kinds of symbols, which are used to represent four kinds of system components.

These are

- Process
- Data stores
- Data flows
- External entity

**Process:**

Process shows the work of the system. Each process has one or more data inputs and produce one or more data outputs. Processes are represented by rounded rectangles in Data Flow Diagram. Each process has a unique name and number. This name and number appears inside the rectangle that represents the process in a Data Flow Diagram.

**Data Stores:**

A data stores is a repository of data. Processes can enter data, into a store or retrieve the data from the data store. Each data has a unique name.

**Data Flows:**

Data flows show the passage of data in the system and are represented by lines joining system components. An arrow indicates the direction of flow and the line is labelled by name of the dataflow.

**External Entity:**

External entities are outside the system but they either supply input data into the system or use other systems output. They are entities on which the designer has control. They may be an organizations customer or other bodies with which the system interacts. External entities that supply data into the system are sometimes called source. External entities that use the system data are sometimes called sinks. These are represented by rectangles in the

Data Flow Diagram.

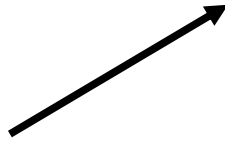
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Basic data flow diagram symbols are.....

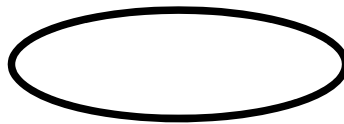
- A Square defines a source (originator) or destination of a system data:



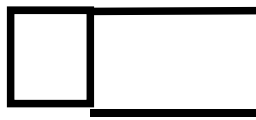
- An Arrow identifies data flow. It is a pipeline through which information flows:



- A Circle represents a process that transforms incoming data flow(s) into outgoing data flow(s):



- An Open Rectangle is a data store:



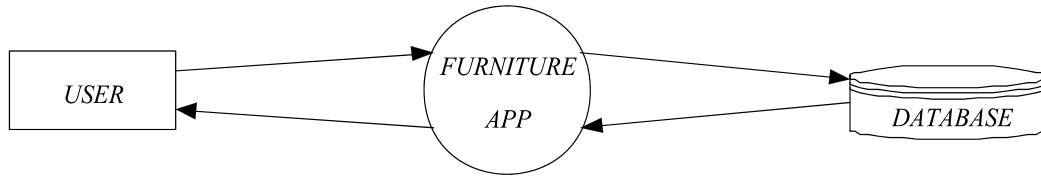
**Four steps are commonly used to construct a DFD:**

- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and left to right.
- When a process is exploded in to lower level details they are numbered.
- The names of data stores, sources and destinations are written in Capital letters.



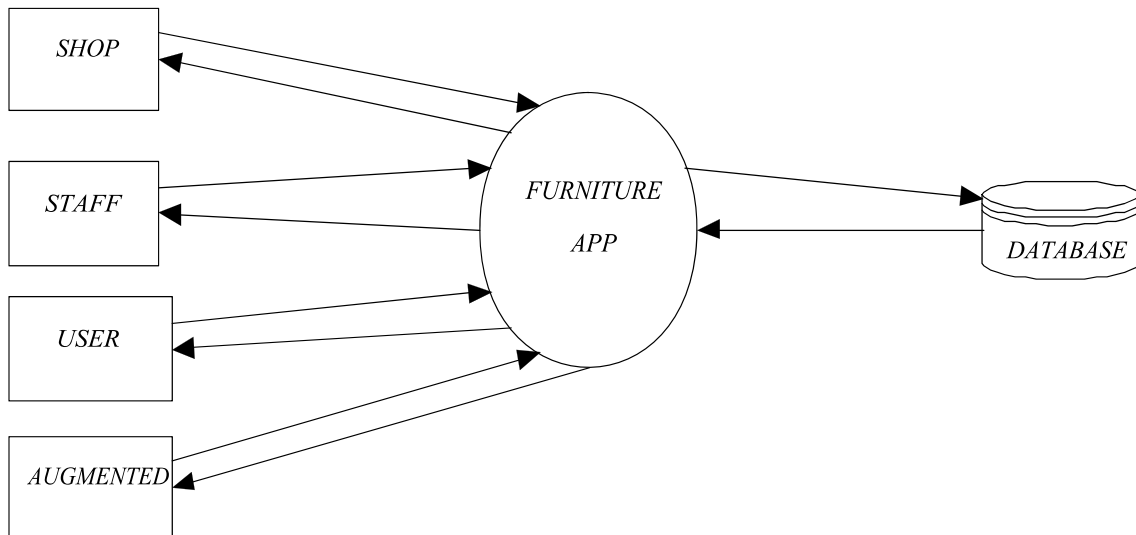
*DFD Level-0*

***LEVEL 0***



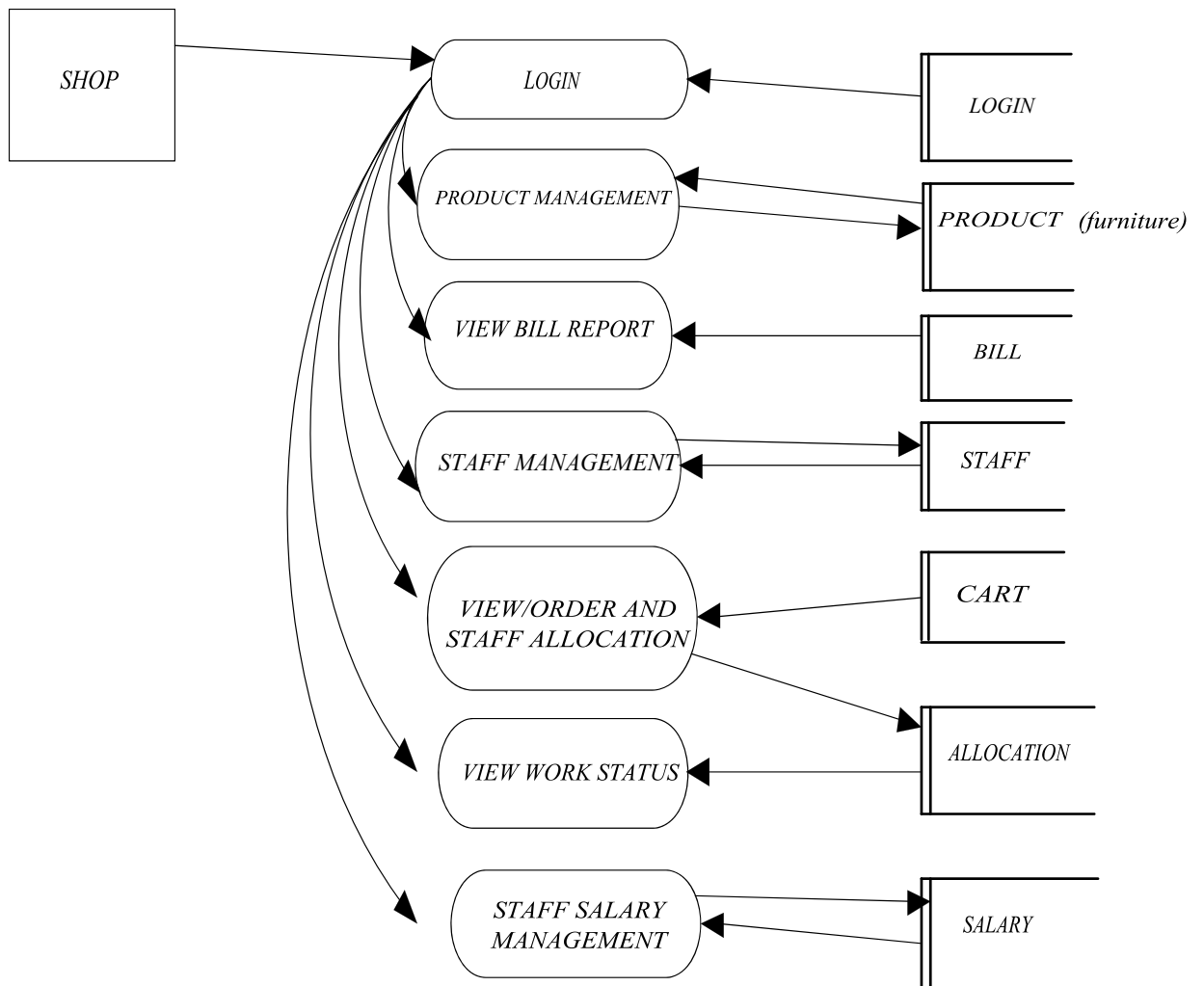
*DFD Level-1*

***LEVEL 1***

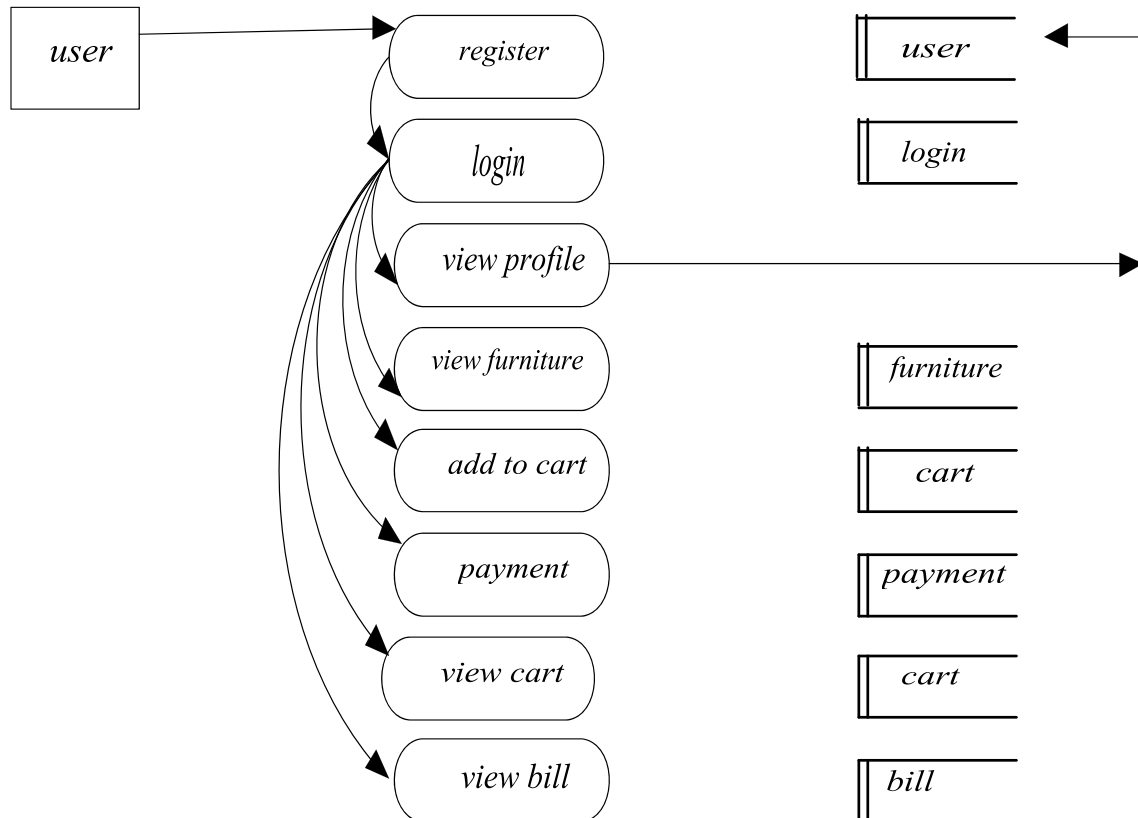


*DFD Level-1.1 Shop*

***LEVEL 1.1***

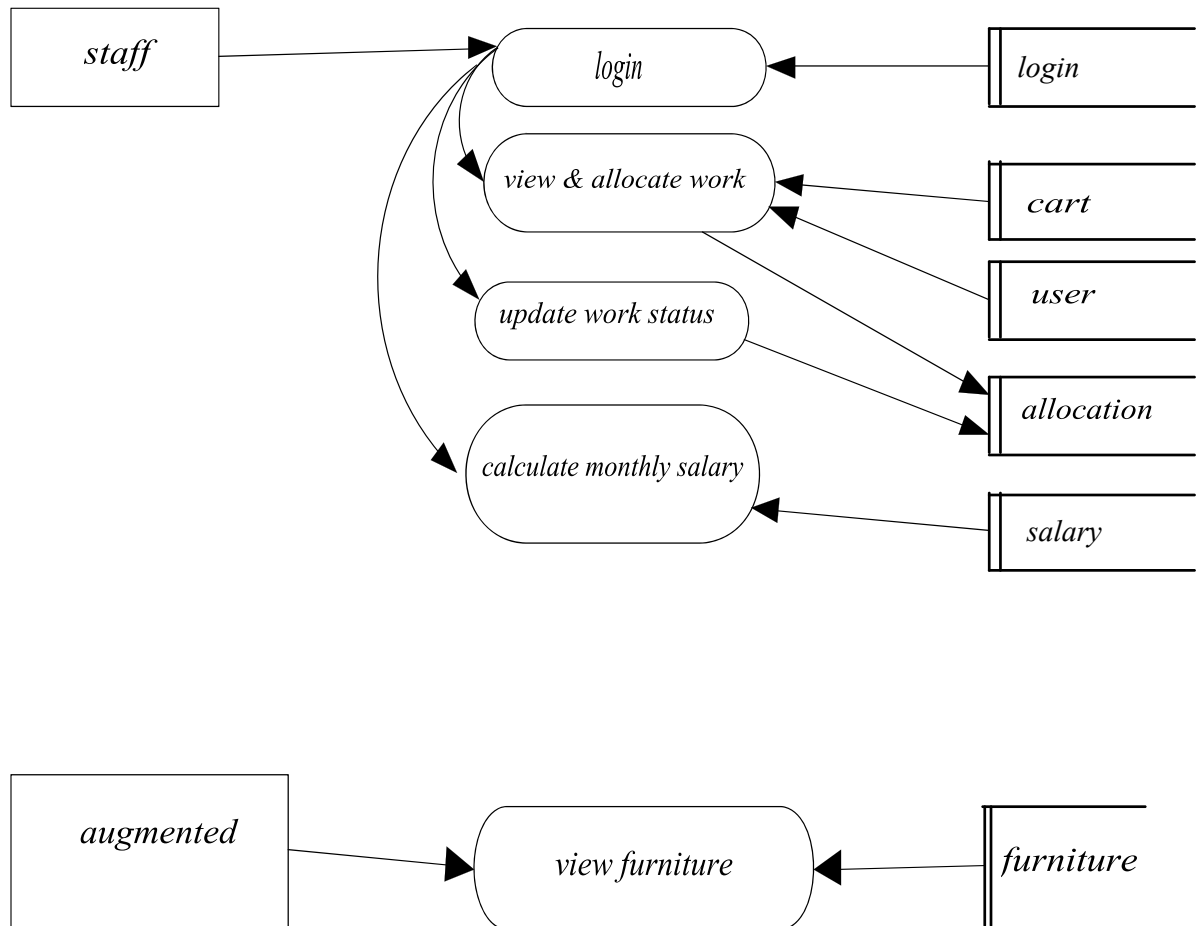


**LEVEL 1.2**



*DFD Level-1.3 STUDENT*

***LEVEL 1.3***



### 3.6 ER Diagram

An ER diagram is a diagram that helps to design databases in an efficient way. It is a data model for describing the data or information. It is a visual representation of data that describes how data is related to each other. The main components of ER models are entities, attributes and the relationships that can exist among them.

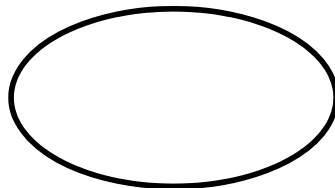
#### Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.



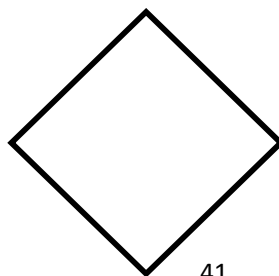
#### Attribute

Attributes are properties of entities. Attributes are represented by means of ellipses. Every ellipse represents one attribute and is directly connected to its entity (rectangle).

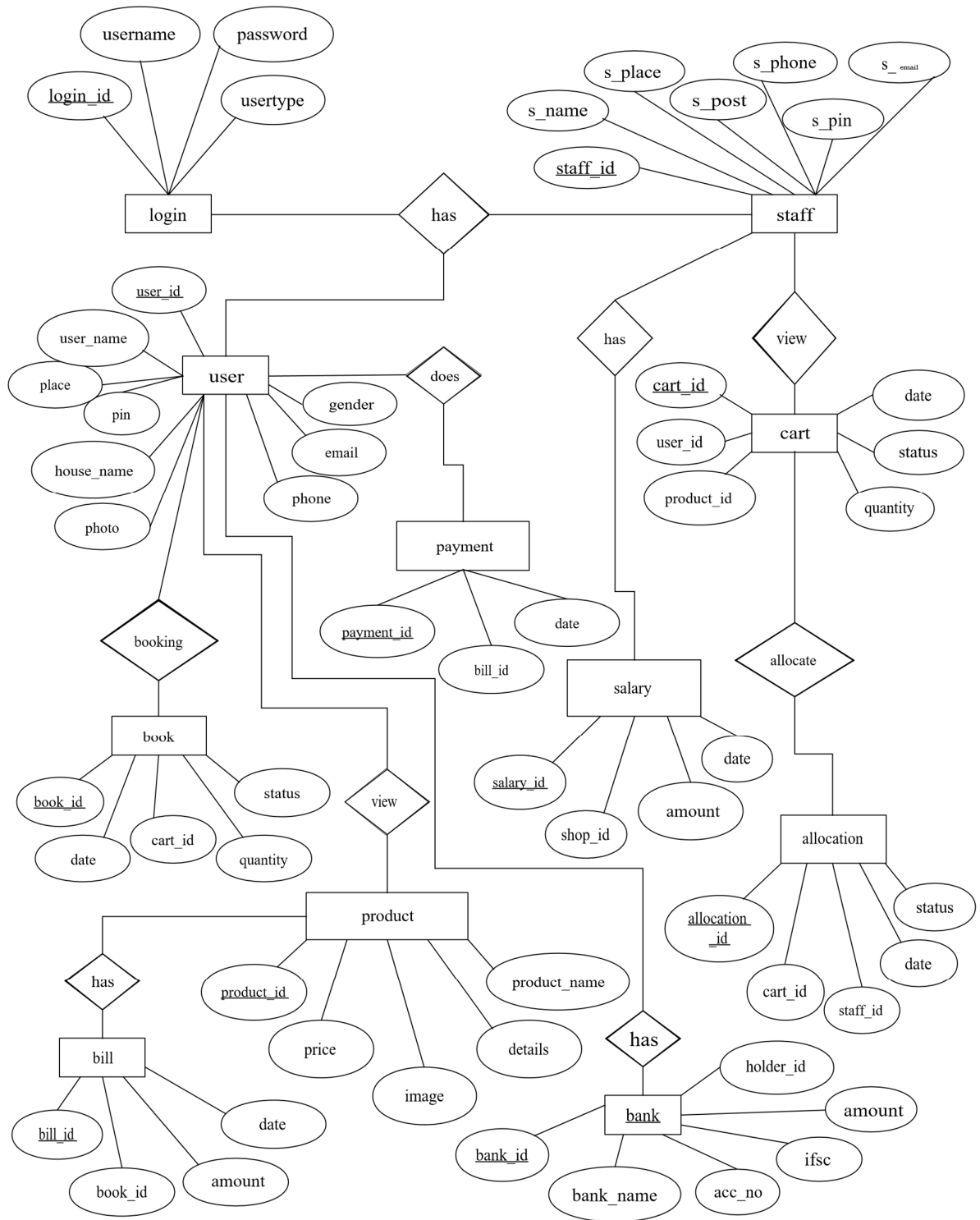


#### Relationship

Relationships are represented by diamond shaped box. Name of the relationship is written in the diamond box. All entities (rectangles), participating in relationship, are connected to it by a line.



## *Architectural design*



## **4.CODING**

## **4.1 INPUT INTERFACE**

Input design is a part of overall system design, which requires very careful attention. If data going into the system is correct, then the processing and output will magnify these errors. Thus the designer has a number of clear objectives in the different stages of input design.

- To produce a cost effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that input is acceptable to and understand by the user.

## **4.2 OUTPUT INTERFACE**

At the beginning of the output design various types of outputs such as external, internal, operational and interactive and turn around are defined. Then the format, content, location, frequency, volume and sequence of the outputs are specified. The content of the output must be defined in detail. The system analysis has two specific objectives at this stage.

- To interpret and communicate the results of the computer part of a system to the users in a form, which they can understand, and which meets their requirements.
- To communicate the output design specifications to programmers in a way in which it is unambiguous, comprehensive and capable of being translated into a programming language.

## **4.3 SOFTWARE DESCRIPTION**

### **4.3.1 HTML**

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML



specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

HTML files are written in ASCII text, so the user can use any text editor to create his/her web page, though a browser of one sort or another is necessary to view the web page. HTML is case insensitive with its language commands. The characters within the document, however, are case sensitive. The language consists of various "tags" which are known as elements. These allow the browser to understand (and put into the desired/specified format) the layout, background, headings, titles, lists, text and/or graphics on the page. The elements are classified according to their function in the HTML document. There are head elements and body elements. The head elements identify properties of the entire document, while body elements actually mark text as content and show a change in the appearance in one way or another. Most elements have a beginning and an ending which encompass the text the user wishes to mark with the tag. All HTML documents must begin with the element and end with the element. Some of the other elements which may be used are tags to create lists-- both ordered lists as well as unordered lists. The user may also create larger or smaller, bolder, italicized, or underlined text. Attributes may be used along with the elements. These perform functions such as placement of text, indication of the source files of images, and identification of links to the document or part of the document.

#### **4.3.2 CSS**

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages, user interfaces for web applications, and user interfaces for many mobile applications. CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colours, and fonts.

## Advantages of CSS

- CSS saves time – you can write CSS once and then reuse same sheet in multiple HTML pages.

You can define a style for each HTML element and apply it to as many Web pages as you want.

- Pages load faster – If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it

to all the occurrences of that tag. So less code means faster download times.

- Easy maintenance – To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- Superior styles to HTML – CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- Multiple Device Compatibility – Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- Global web standards – Now HTML attributes are being deprecated and it is being recommended to use CSS. So it is a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.
- Offline Browsing – CSS can store web applications locally with the help of an offline cache.

Using of this, we can view offline websites. The cache also ensures faster loading and better overall performance of the website.

- Platform Independence – The Script offer consistent platform independence and can support latest browsers as well.

## 4.3.3 JAVASCRIPT

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the

user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript.

The General-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

Advantages of JavaScript:

- Less server interaction – you can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.
- Immediate feedback to the visitors – They don't have to wait for a page reload to see if they have forgotten to enter something.
- Increased interactivity – you can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.
- Richer interfaces – you can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

#### **4.3.4 MySQL**

MySQL is an open-source relational database management system (RDBMS). MySQL is released under an open-source license. So you have nothing to pay to use it. MySQL is a very powerful program in its own right.

It handles a large subset of the functionality of the most expensive and powerful database packages. It uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.

It works very quickly and works well even with large data sets. It is very friendly to PHP, the most appreciated language for web development. It supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB). It is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

## Major features as available in MySQL 5.6

- A broad subset of ANSI SQL 99, as well as extensions.
- Cross-platform support.
- Stored procedures, using a procedural language that closely adheres to SQL/PSM.
- Triggers.
- Cursors.
- Updatable views.
- Online DDL when using the InnoDB Storage Engine.
- Information schema.
- Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
- A set of SQL Mode options to control runtime behaviour, including a strict mode to better adhere to SQL standards.
  
- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine.
- Transactions with savepoints when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
- ACID compliance when using InnoDB and NDB Cluster Storage Engines.
- SSL support → Query caching → Sub-SELECTs (i.e. nested SELECTs) .
- Built-in replication support (i.e., master-master replication and master-slave replication) with one master per slave, many slaves per master.
- Multi-master replication is provided in MySQL Cluster, and multimaster support can be added to unclustered configurations using Galera Cluster.
- Full-text indexing and searching.
- Embedded database library.
- Unicode support.
- Partitioned tables with pruning of partitions in optimizer.
- Shared-nothing clustering through MySQL Cluster.
- Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.
- Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.

- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

## **Advantages**

MySQL database server has lots of advantages over its competitors. Some of these advantages have been explained below.

- Open Source and Cost Effective:

The best thing about MySQL server is that this is open source and it has a free version as well.

By open source software, we mean that the code of the software is available and anyone can tailor it according to his requirement. Companies prefer MySQL because they don't have to pay anything for this excellent product.

- Portability:

MySQL is cross platform database server. MySQL can be run on a variety of platforms including Windows, OS2, Linux and Solaris. Portability of MySQL server makes it suitable for applications that target multiple platforms particularly web application. MySQL contains API for almost all the major programming languages and can be easily integrated with the languages like PHP, C++, Perl, C, Python and ruby. In fact, MySQL is a part of the famous LAMP (Linux Apache MySQL PHP) server stack which is used worldwide for web application development.

- Seamless Connectivity:

Various secure and seamless connection mechanisms are available in order to connect with MySQL server. These connections include named pipes, TCP/IP sockets and UNIX Sockets.

- Rapid Development and Continuous Updates:

Being an open source product, MySQL has a very large developer community which releases regular patches and updates for MySQL. Several database templates have been developed which can be readily used and modified resulting in rapid application development.

- Security:

MySQL server databases are extremely secure and all the data access scenarios are protected via password and good thing about these passwords is that they are stored

in encrypted form and it is not easy to break these advanced and complex encryption algorithms.

#### 4.3.5 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

#### Major features of python

- Easy to code
- Free and Open Source
- Object-Oriented Language
- GUI Programming Support
- High-Level Language
- Extensible feature
- Python is **Portable** language

- Python is Integrated language

### **Advantages**

- Extensive support libraries
- Integration feature
- Improved productivity
- Easy to learn and write
- Vast library support
- Free and open source

### **4.3.6 Flask**

Flask is a web application framework written in Python. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Pocco. Flask is based on the Werkzeug WSGI toolkit and the Jinja2 template engine. Both are Pocco projects.

It is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier. It gives developers flexibility and is a more accessible framework for new developers since you can build a web application quickly using only a single Python file. Flask is also extensible and doesn't force a particular directory structure or require complicated boilerplate code before getting started.

## **5.CODING PAGES**



## 5.1 Shop Page

```
@app.route('/')
def login():
    return render_template("index.html")

@app.route('/post_login',methods=['post'])
def post_login():
    username=request.form['textfield']
    password=request.form['textfield2']
    db=Db()
    res=db.selectOne("select * from login where username='"+username+"' and
password='"+password+"'")
    if res is not None:
        if res['usertype']=='shop':
            session['lg']='lin'
            return redirect("/shop_home")
        elif res['usertype']=='staff':
            session['lid']=res['login_id']
            return redirect("/staff_home")
        else:
            return("invalid")
    else:
        return("invalid")

@app.route('/add_product')
def add_product():
    if session['lg']=='lin':
        return render_template('shop/add_product.html')
    else:
        return redirect('/')

@app.route('/post_add_product',methods=['post'])
def post_add_product():

    if session['lg']=='lin':

        pname=request.form['textfield']
        image=request.files['fileField']
        stock=request.form['textfield2']
        price=request.form['textfield3']
        details=request.form['details']
        db=Db()
        date=datetime.datetime.now().strftime("%d%m%y-%H%M%S")
```

```

        image.save(static_path + "pic\\"+date+'.jpg')
        path="/static/pic/"+date+'.jpg'
        db.insert("insert into product(product_name,price,image,details,stock)
values('"+pname+"','"+price+"','"+str(path)+"','"+details+"','"+stock+"')")
        return '<script>alert("success");window.location="/shop_home"</script>'
    else:
        return redirect('/')

```

```

@app.route('/add_salary/<sid>')
def add_salary(sid):
    if session['lg']== 'lin':

        return render_template('shop/add_salary.html', sid=sid)
    else:
        return redirect('/')

```

```

@app.route('/post_add_salary',methods=['post'])
def post_add_salary():
    if session['lg']== 'lin':

        sid=request.form['hid']
        amount=request.form['textfield']
        db=Db()
        db.insert("insert into salary(staff_id,amount,date)
values('"+str(sid)+"','"+amount+"',curdate())")
        return '<script>alert("success");window.location="/shop_home"</script>'
    else:
        return redirect('/')

```

```

@app.route('/add_staff')
def add_staff():
    if session['lg']== 'lin':

        return render_template('shop/add_staff.html')
    else:
        return redirect('/')

```

```

@app.route('/post_add_staff', methods=['post'])
def post_add_staff():
    if session['lg']== 'lin':

```

```

name=request.form['textfield']
place=request.form['textfield2']
post=request.form['textfield3']
pin=request.form['textfield4']
contact=request.form['textfield5']
email=request.form['textfield6']
password=random.randint(1000, 9999)
db=Db()
res=db.insert("insert into login(username, password, usertype) values('"+email+"',
'"+str(password)+"', 'staff')")
db.insert("insert into staff(staff_id,s_name,s_place,s_post,s_pin,s_phone,s_email)
values('"+str(res)+"', '"+name+"', '"+place+"', '"+post+"', '"+pin+"', '"+contact+"', '"+emai
l+"')")

    return '<script>alert("success");window.location="/shop_home"</script>'
else:
    return redirect('/')

@app.route('/allocate_staff/<cid>')
def allocate_staff(cid):
    if session['lg']=='lin':

        db=Db()
        res=db.select("select * from staff")

        return render_template('shop/allocate_staff.html',data=res,cid=cid)
    else:
        return redirect('/')

@app.route('/post_allocate_staff', methods=['post'])
def post_allocate_staff():
    if session['lg']=='lin':

        cid=request.form['cid']

        staff=request.form['select']
        db=Db()
        db.insert("insert into allocation VALUES
('"+cid+"', '"+staff+"', 'allocated',curdate()) ")
        db.update("update cart set status='allocated' where cart_id='"+cid+"'")
        return "<script>alert('Order allocated');window.location='/view_order';</script>"
    else:
        return redirect('/')

@app.route('/edit_product/<pid>')
def edit_product(pid):
    if session['lg']=='lin':

        db=Db()

```

```

    res=db.selectOne("select * from product where product_id='"+str(pid)+"'")
    return render_template('shop/edit_product.html',pid=pid,data=res)
else:
    return redirect('/')

@app.route('/post_edit_product', methods=['post'])
def post_edit_product():
    if session['lg']=='lin':

        pid=request.form['hid']
        pname=request.form['textfield']
        image=request.files['fileField']
        stock=request.form['textfield2']
        price=request.form['textfield3']
        details=request.form['textfield4']
        date = datetime.datetime.now().strftime("%d%m%y-%H%M%S")
        image.save(static_path + "pic\\" + date + '.jpg')
        path = "/static/pic/" + date + '.jpg'
        db=Db()
        res=db.selectOne("select * from product where product_name='"+pname+"' and
image='"+str(path)+"' and stock='"+stock+"' and price='"+price+"' and
details='"+details+"' and product_id='"+str(pid)+"'")
        if res is not None:
            return '<script>alert("no change");window.location="/view_product"</script>'
        else:
            if request.files!="none":
                if image.filename!="":
                    db.update("update product set product_name='"+pname+"',
image='"+str(path)+"', stock='"+stock+"',price='"+price+"',details='"+details+"' where
product_id='"+str(pid)+"'")
                    return
            '<script>alert("success");window.location="/view_product"</script>'
            else:
                db.update("update product set product_name='"+pname+"' ,stock='"+
stock + "',price='"+ price + "',details='"+ details + "' where
product_id='"+str(pid)+"'")
                return
            '<script>alert("success");window.location="/view_product"</script>'
            else:
                db.update("update product set product_name='"+pname+"' ,stock='"+ stock
+ "',price='"+ price + "',details='"+ details + "' where product_id='"+ str(pid) + "'")
                return '<script>alert("success");window.location="/view_product"</script>'
            else:
                return redirect('/')

@app.route('/edit_staff/<sid>')
def edit_staff(sid):
    if session['lg']=='lin':

        db=Db()

```

```

        res=db.selectOne("select * from staff where staff_id='"+str(sid)+"'")
        return render_template('shop/edit_staff.html',sid=sid,data=res)
    else:
        return redirect('/')

@app.route('/post_edit_staff', methods=['post'])
def post_edit_staff():
    if session['lg']=='lin':

        sid=request.form['hid']
        name=request.form['textfield']
        place=request.form['textfield2']
        post=request.form['textfield3']
        pin=request.form['textfield4']
        contact=request.form['textfield5']
        email=request.form['textfield6']
        db=Db()
        db.update("update staff set
s_name='"+name+"',s_place='"+place+"',s_post='"+post+"',s_pin='"+pin+"',s_phone='
"+contact+"',s_email='"+email+"' where staff_id='"+str(sid)+"' ")
        return '<script>alert("success");window.location="/view_staff"</script>'
    else:
        return redirect('/')

@app.route('/shop_home')
def shop_home():
    if session['lg']=='lin':

        return render_template('shop/index.html')
        #return render_template('shop/shop_home.html')
    else:
        return redirect('/')

@app.route('/view_order')
def view_order():
    if session['lg']=='lin':

        db=Db()
        res=db.select("select product.*, cart.*, user.*, product.price*cart.quantity as
amount from product,cart,user where product.product_id=cart.product_id and
user.user_id=cart.user_id ")
        return render_template('shop/view_order.html',data=res)
    else:
        return redirect('/')

@app.route('/view_product')
def view_product():
    if session['lg']=='lin':

```

```

    db=Db()
    res=db.select("select * from product")
    return render_template('shop/view_product.html',data=res)
else:
    return redirect('/')
@app.route('/view_salary')
def view_salary():
    if session['lg']=='lin':

        db=Db()
        res=db.select("select * from salary,staff where salary.staff_id=staff.staff_id")
        return render_template('shop/view_salary.html',data=res)
    else:
        return redirect('/')

@app.route('/view_staff')
def view_staff():
    if session['lg']=='lin':

        db=Db()
        res=db.select("select * from staff")
        return render_template('shop/view_staff.html',data=res)
    else:
        return redirect('/')

@app.route('/view_work_status')
def view_work_status():
    if session['lg']=='lin':

        db=Db()
        res = db.select("select product.*, cart.*, user.*, allocation.*, staff.*,
        cart.quantity*product.price as amount from product,cart,user,allocation,staff where
        product.product_id=cart.product_id and user.user_id=cart.user_id and
        allocation.cart_id=cart.cart_id and staff.staff_id=allocation.staff_id ")
        return render_template('shop/view_work_status.html',data=res)
    else:
        return redirect('/')

@app.route('/delete_product/<pid>')
def delete_product(pid):
    if session['lg']=='lin':

        db=Db()
        db.delete("delete from product where product_id='"+str(pid)+"'")
        return redirect('/view_product')
    else:
        return redirect('/')

@app.route('/delete_staff/<sid>')

```

```

def delete_staff(sid):
    if session['lg']=='lin':

        db=Db()
        db.delete("delete from staff where staff_id='"+str(sid)+"'")
        return redirect('/view_staff')
    else:
        return redirect('/')

@app.route('/delete_salary/<sid>')
def delete_salary(sid):
    if session['lg']=='lin':

        db=Db()
        db.delete("delete from salary where salary_id='"+str(sid)+"'")
        return redirect('/view_salary')
    else:
        return redirect('/')

```

## 5.2 Staff Page

```
@app.route('/staff_home')
def staff_home():
    if session['lg']!= 'lin':

        return render_template('staff/staff_home.html')
    else:
        return redirect('/')

@app.route('/view_allocate_work')
def view_allocate_work():
    if session['lg']!= 'lin':

        db=Db()
        res=db.select("select product.*, cart.*, user.*, allocation.*,
        cart.quantity*product.price as amount, cart.status as cstatus from
        product, cart, user, allocation where product.product_id=cart.product_id and
        user.user_id=cart.user_id and allocation.cart_id=cart.cart_id and
        allocation.staff_id='"+str(session['lid'])+"'")
        return render_template('staff/view_allocate_work.html', data=res)
    else:
        return redirect('/')

@app.route("/staff_out_for_delivery/<cid>/<aid>")
def staff_out_for_delivery(cid, aid):
    if session['lg']!= 'lin':

        db=Db()
        db.update("update cart set status='out_for_delivery' where cart_id='"+cid+"'")
        db.update("update allocation set status='out_for_delivery' where
        allocation_id='"+aid+"'")
        return redirect("/view_allocate_work")
    else:
        return redirect('/')

@app.route("/staff_delivered/<cid>/<aid>")
def staff_delivered(cid, aid):
    if session['lg']!= 'lin':

        db=Db()
        db.update("update cart set status='delivered' where cart_id='"+cid+"'")
        db.update("update allocation set status='delivered' where allocation_id='"+aid+"'")
        return redirect("/view_allocate_work")

    else:
```



```

    return redirect('/')

@app.route('/view_product1')
def view_product1():
    if session['lg']!='lin':

        return render_template('staff/view_product1.html')
    else:
        return redirect('/')

@app.route('/staff_view_salary')
def staff_view_salary():
    if session['lg']!='lin':

        db=Db()
        res=db.select("select * from salary where staff_id='"+str(session['lid'])+"'")
        return render_template('staff/view_salary.html', data=res)
    else:
        return redirect('/')

@app.route('/logout')
def logout():
    session.clear()
    session['lg']="''
    return redirect('/')

```

## **6. SYSTEM TESTING**

## 6.1 TESTING AND EVALUATION

Testing is a process of executing a program with the intent of finding an error. Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design and coding. Testing includes verifications of the basic logic of each program and verification that the entire system works properly. Testing demonstrates that software functions appear to be working according to specification. In addition, data collected as testing is conducted provides a good indication of software quality as a whole. The debugging process is the most unpredictable part of testing process. Testing begins at the module level and works towards the integration of the entire computer-based system. Testing and debugging are different activities, but any testing includes debugging strategy for software testing must accommodate low level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system function, against customer requirements. No testing is complete without verification and validation part. The goals of verification and validation activities are to assess and improve the quality of work products generated during the development and modification of the software. There are two types of verification: life cycle verification and formal verification. Life cycle verification is the process of determining the degree to which the products of the given phase of the development cycle fulfil the specification established during the prior process. Formal verification is the rigorous mathematical demonstration that source code conforms to its specifications. Validation is a process of evaluating software at the end of the software development process to determine conformance with the requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The primary objectives, when we test software are the following:

- Testing is a process of executing with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.
- A successful test is one uncovers undiscovered errors.

Thus, testing plays a very critical role in determining the reliability and efficiency of the software and hence is very important stage in software development. Tests are to be conducted on the software to evaluate its performance under a number of conditions. Ideally, it should do so at the level

of each module and also when all of them are integrated to form the completed system. Software testing is done at different levels.

## **6.2 TESTING STRATEGIES**

A strategy for software testing integrates software test case design method into a well-planned series of steps that result in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote a customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project progresses.

**The general characteristics of software testing strategies are:**

- Testing begins at the component level and works “outward” toward the integration of the entire computer system.
- Different testing techniques are appropriate at different points in time.

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

A strategy must provide guidance for the practitioner and set of milestones for the manager. Because the step on the test strategy occurs at a time when deadline pressure begins to rise, progress must be measurable and problem must surface as early as possible.

The software team’s approach to testing is defining a plan that describes an overall strategy and a procedure that defines specific testing steps and tests that will be conducted. In the proposed system, if the administrator makes any attempt to login to the application without entering his password, then the system will not allow the user to login to the application.

## **6.3 TESTING TECHNIQUES**

The various testing techniques are given below:

### **6.3.1 WHITE BOX TESTING**

White-box testing is also called as glass-box testing, is a test case design method that goes to the control structure of the procedural design to derive

test cases. Using white box testing methods, the software engineer can derive test cases that,

- ✓ Guarantee that all independent paths within a module have been exercised at Least once.
- ✓ Exercise all logical decision on their true and false sides.
- ✓ Execute all loops at their boundaries and within their operational sides.
- ✓ Exercise internal data structure to ensure their validity.

White box testing was successfully conducted on our system. All independent paths within a module have been executed at least once and all logical decisions have been exercised on their true and false sides.

### **6.3.2 BLACK BOX TESTING**

Black-box testing is also called as behavioural testing, focuses on the functional requirement of the software. It is a complimentary approach that is likely uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories.

- ✓ Incorrect or missing functions.
- ✓ Interface errors.
- ✓ Error on data structures or external database access.
- ✓ Behaviour or performance errors.
- ✓ Initialization and termination errors.

Black box testing was successfully conducted on your system. The system was divided into a number of modules and testing was conducted on each module. We have tested the system for incorrect or missing functions, interface and performance errors.

### **6.3.3 UNIT TESTING**

Unit testing comprises the set of tests performed by an individual programmer prior to the integration of the system. Testing removes residual bugs and improves the reliability of the system.

Testing allows the developer to find out the design faults if any, and enable correction if needed. Exhaustive unit testing has to be carried out to ensure the validity of the data. In order to successfully test the entire package, unit testing is carried out. Each module was tested as when it was developed. Thus it proved easier to conduct minute testing operation and correct them then and there.

### **6.3.4 INTEGRATION TESTING**

Bottom-up integration is the traditional strategy used to integrate the component of a software system into a functional whole. Bottom-up integration consists of unit testing, followed by subsystem testing and followed by testing of the entire system. Unit testing has the goal of discovering errors in the individual parts of the system.

Parts are tested in isolation from one another in an artificial environment known as “Test Harness”, which consists of driver programs and data necessary to exercise the modules. Unit testing should be as exhaustive as possible to ensure that each representative case handled by each module has been tested. Unit testing is eased by a system structure that is composed of small loosely coupled modules.

Both control and data interfaces must be tested. Large software system may require several levels of subsystem testing. Lower level subsystems are successively combined to form higher level subsystems. In most software systems, exhaustive testing of subsystem capabilities is not feasible due to the combination complexity of the module interfaces. Therefore, test cases must be carefully chosen to exercise the interfaces in the desired manner.

### **6.3.5 ACCEPTANCE TESTING**

Acceptance testing involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements. It is

not unusual for two sets of acceptance test to be run, those developed by the quality group and those developed by the customer.

In addition to the functional and performance tests, stress tests are performed to determine the limitation of the system. For example, a compiler might be tested to determine the effect of the symbol table overflow, or real-time system might be tested to determine the effect of simultaneous arrival of numerous high priority interrupts.

#### **6.3.6 OUTPUT TESTING**

Output testing of the proposed system is important since no system could be useful if it does not produce the required output.

The output format on the screen is found to be correct, as the format was designed in the system design phase according to the user needs. For the hard copy also the output comes out as the specified requirements by

the user. Hence output testing doesn't result in any correction on the system.

## **7.SYSTEM IMPLEMENTATION AND DEPLOYMENT**

Implementation is the process of deploying the new system in the operational environment. Proper implementation is essential to provide a reliable system to meet the organizational requirement. There are four types of implementation methods. They are Direct Changeover, Phased Implementation, Parallel Run and Pilot Approach. The most commonly used implementation methods are Pilot Approach and Parallel Run.

The system which is developed as a web application hence the other functions as normal application, as usual some web development technologies are used in the implementation of the project. The language I selected to program this software is PHP. The reason I selected PHP is that is a simple and powerful language that especially developed to create web application.

Technologies used in the development of the software are:

- ✓ Programming language – Python
- ✓ DBMS – MySQL server
- ✓ Development tool – Pycharm
- ✓ Development platform – Windows 10

The front end is HTML, and CSS and back end is MySQL Server and Python. The system developed on Pycharm in Windows 10 operating system.



## **8. CONCLUSION**

Traditional shopping industries have been very successful in the past, however today's world requires a convenient way of doing things. May it be shopping or getting a cab. This change may be due to a different generation of individuals as a customer base for most of the industries, or due to the technological revolution in the past few years. Thus the Furnished application fits perfectly in this world where convenience is paramount for the customers. We are able to achieve the goal of saving time, costs and hassles by the use of AR.

### **8.1 FUTURE ENHANCEMENT**

As a future venture, it is suggested to make some changes to provide more services and to provide information at right time in right manner.

The current system is designed to detect helmet and mask violations in the college premises. It can only detect the violations taking place within the college.

In the future the system can be extended to detect violations in a larger area like the village or Panchayat or even more. More features like detecting whether the students have worn ID cards or not can also be added to the system. All the functions have been done carefully and successfully in the software, and if any development is necessary in future, it can be done without affecting the design by adding additional modules to the system.

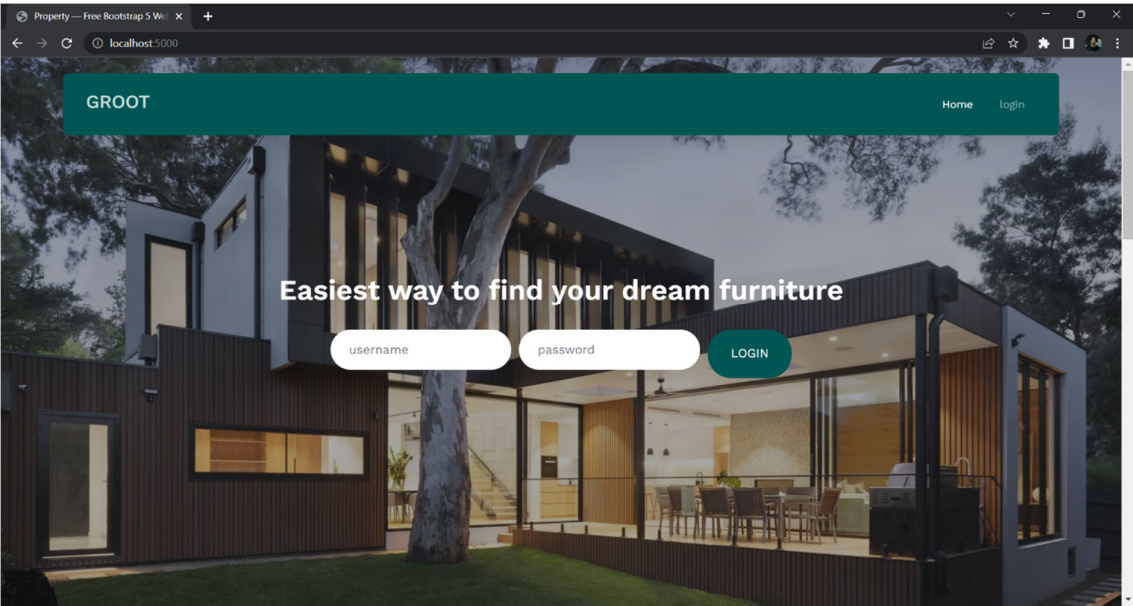
## 9. REFERENCE

- Google.com
- Youtube

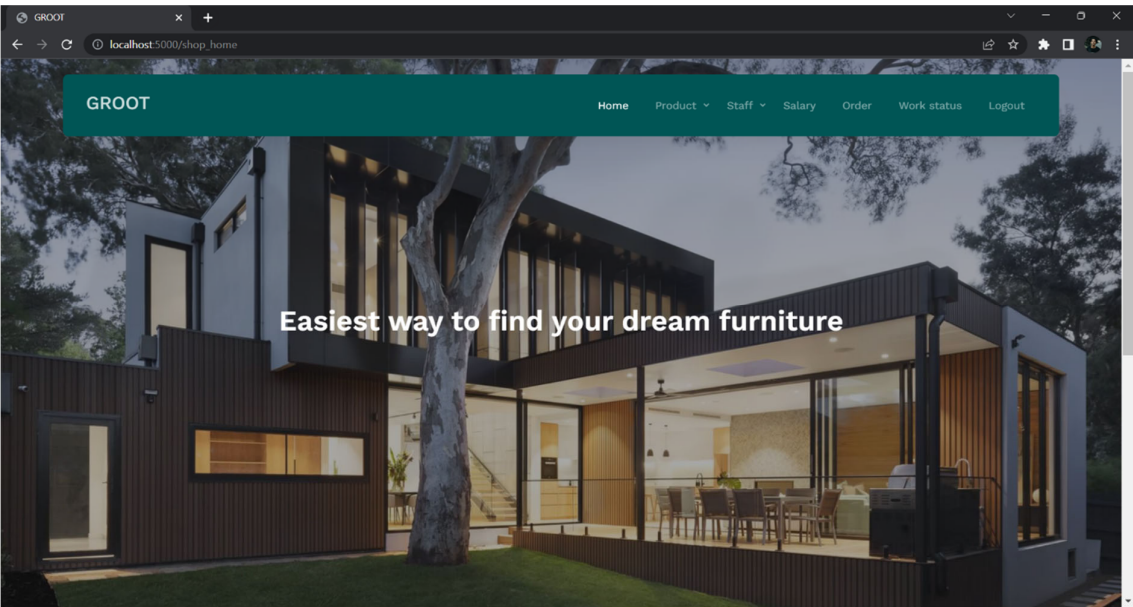
## **10. APPENDIX**

10.1 Screenshots

10.1.1 Login page:



10.1.2 Shop page:



## Add product

Product Name

Image

Choose File

No file chosen





stock

price

details

Add

## View product

#	Product Name	Image	Stock	Price	Details	
1	Sofa		7	25000	Primary Material - Polyester (Fabric)Secondary Material - Rubber WoodFilling Material - Foam	<div>edit</div> <div>delete</div>
2	table		20	5000	Compact Design - Sleek Style - Sturdy Construct - Easy To Move - Low Maintenance	<div>edit</div> <div>delete</div>
3	artichoke		15	8000	eco-friendly leather, durable	<div>edit</div> <div>delete</div>
4	couch		20	9000	material - wood, single seater	<div>edit</div> <div>delete</div>

### Add staff

GROOT

localhost:5000/add\_staff#about

name

place

post

pin

contact

email

add

### View staff

GROOT

localhost:5000/view\_staff#about

#	staff name	address	contact	email	
1	Anupam	Thalassery Champad 670694	9188650352	4nupamr@gmail.com	<div>edit</div> <div>delete</div> <div>add salary</div>
2	Mackey	Vadakara Beach post 673103	8590094668	mackeyabdulrahim243@gmail.com	<div>edit</div> <div>delete</div> <div>add salary</div>
3	Ashwini	Thalassery kuyyali 670693	9988776665	ashwinijithesh@gmail.com	<div>edit</div> <div>delete</div> <div>add salary</div>

## View Salary

#	date	staff name	salary	
1	2023-03-20	Anupam	30000	delete
2	2023-03-20	Mackey	25000	delete
3	2023-03-20	Ashwini	20000	delete
4	2023-03-20	anumod	20000	delete

**CONTACT**  
thalassery kannur 670101  
+91 9188650352  
+91 6235043359  
groot23@gmail.com

**SOURCES**  
[About us](#)  
[Services](#)  
[Vision](#)  
[Mission](#)  
[Terms](#)

**LINKS**  
[Our Vision](#)  
[About us](#)  
[Contact us](#)  
[Partners](#)  
[Business](#)  
[Careers](#)  
[Blog](#)  
[FAQ](#)

[@](#)[@](#)[f](#)[in](#)[@](#)[@](#)

## View order

#	user info	product name	price	quantity	amount	
1	Anumod Kannur 670001 7012159950	tv furniture	40000	1	40000	allocated
2	Anumod Kannur 670001 7012159950	Sofa	25000	1	25000	added to cart
3	Anumod Kannur 670001 7012159950	artichoke	8000	1	8000	delivered

**CONTACT**  
thalassery kannur 670101  
+91 9188650352  
+91 6235043359

**SOURCES**  
[About us](#)  
[Services](#)  
[Vision](#)

**LINKS**  
[Our Vision](#)  
[About us](#)  
[Contact us](#)  
[Partners](#)  
[Business](#)  
[Careers](#)

### View work status

GROOT

localhost:5000/view\_work\_status#about

#	user info	user address	product info	quantity	bill amount	assinged staff	status
1	Anumod	Kannur 670001 7012159950	tv furniture	1	40000	Ashwini	allocated
2	Anumod	Kannur 670001 7012159950	artichoke	1	8000	anumod	delivered

CONTACT

thalassery kannur 670101

+91 9188650352

+91 6235043359

groot23@gmail.com

SOURCES

About us

Services

Vision

Mission

Terms

Privacy

Partners

Business

Careers

Blog

FAQ

Creative

LINKS

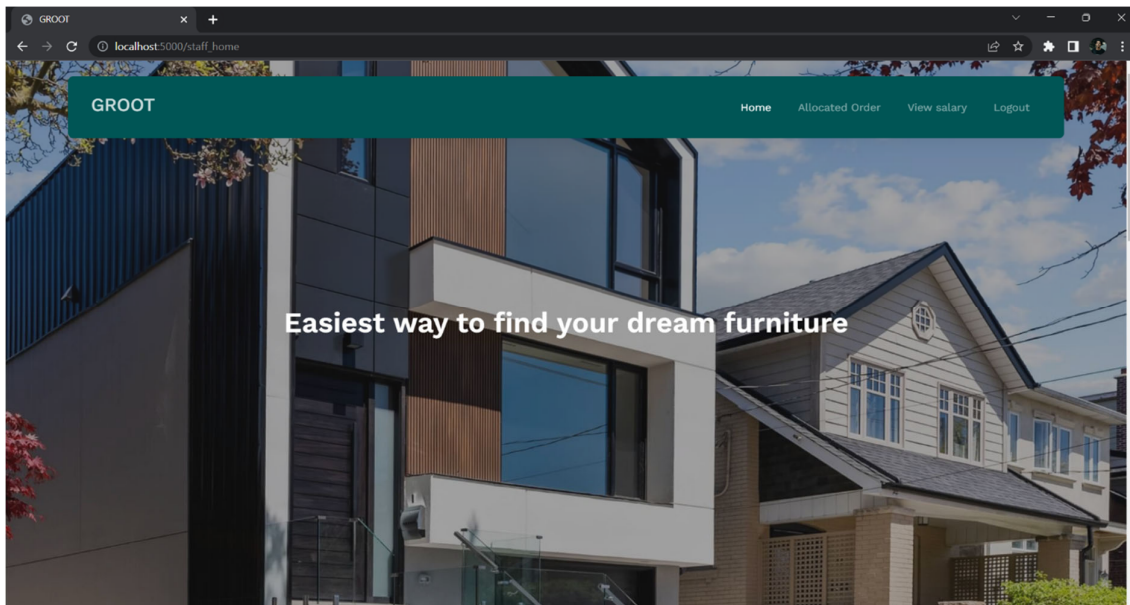
Our Vision

About us

Contact us

### 10.1.3 Staff page:





## Allocated order

#	user info	user address	product info	quantity	bill amount
1	Anumod	Kannur 670001 7012159950	artichoke	1	8000

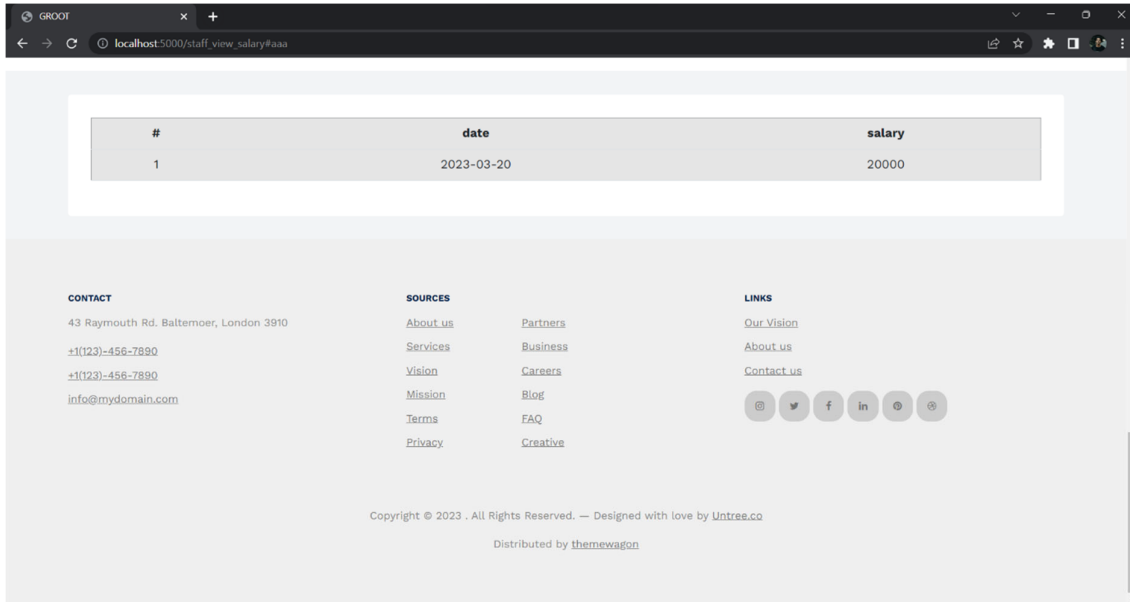
**CONTACT**  
43 Raymouth Rd. Baltemoer, London 3910  
+1(123)-456-7890  
+1(123)-456-7890  
info@mydomain.com

**SOURCES**  
[About us](#)  
[Services](#)  
[Vision](#)  
[Mission](#)  
[Terms](#)  
[Privacy](#)

**LINKS**  
[Our Vision](#)  
[About us](#)  
[Contact us](#)

Copyright © 2023 . All Rights Reserved. — Designed with love by [Untree.co](#)  
Distributed by [themewagon](#)

## View Salary



**A PROJECT REPORT ON**  
**LARVA- LAB ATTENDANCE REGISTRATION VIA**  
**APPLICATION**

*Submitted in partial fulfilment of the requirement for the award of*

**BCA**

**Degree of**

**KANNUR UNIVERSITY**

*Submitted by*

**ADON BESTY**

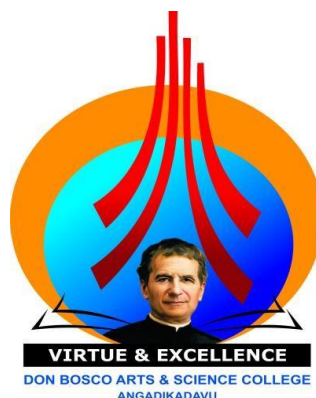
**Reg No: DB20BCAR18**

**SANJO JOSE AUGUSTINE**

**Reg No: DB20BCAR11**

**UNDER THE SUPERVISION AND GUIDANCE OF**

**Mrs. VINEETHA MATHEW**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADVU, KANNUR, KERALA**

**2023**

**A PROJECT REPORT ON**  
**LARVA- LAB ATTENDANCE REGISTRATION VIA**  
**APPLICATION**

*Submitted in partial fulfilment of the requirement for the award of*

**BCA**

**Degree of**

**KANNUR UNIVERSITY**

*Submitted by*

**ADON BESTY**

**Reg No: DB20BCAR18**

**SANJO JOSE AUGUSTINE**

**Reg No: DB20BCAR11**

**UNDER THE SUPERVISION AND GUIDANCE OF**

**Mrs. VINEETHA MATHEW**



**DON BOSCO ARTS & SCIENCE COLLEGE**  
**ANGADIKADVU, KANNUR, KERALA**

**2023**

# KANNUR UNIVERSITY

## DON BOSCO ARTS & SCIENCE COLLEGE

ANGADIKADVU, KANNUR



### CERTIFICATE

Certified that this report titled **LARVA-Lab Attendance Register via Application.** The project work done by **ADON BESTY (Reg.No: DB20BCAR18)** and **SANJO JOSE AUGUSTINE (Reg. No: DB20BCAR11)** under the supervision and guidance, towards partial fulfilment of the requirement forward of the degree of bachelor of computer application (BCA) of the Kannur university.

**Project Guide:**

**Mrs. VINEETHA MATHEW**

Angadikadavu

Date:

**Head of the Department:**

**Mrs. SINDHU P.M**

External Examiner

1.

2.

## **Declaration**

We **ADON BESTY** and **SANJO JOSE AUGUSTINE** sixth semester BCA students of Don Bosco Arts & Science College Angadikadavu, under Kannur University do hereby declare that the project entitled “**LARVA-LAB ATTENDACE REGISTRATION VIA APPLICATION** ” is the original work carried out by us in the sixth semester under the supervision of **Mrs. VINEETHA MATHEW**, Lecture of the Department of BCA, Don Bosco Arts & Science College, Angadikadavu, in partial fulfilment of the requirement for the award of the degree Bachelor of Computer Application, Kannur University.

ANGADIKADAVU

ADON BESTY

DATE :

SANJO JOSE AUGUESTINE

## ACKNOWLEDGEMENT

First of all we thank the lord almighty for his immense grace and blessings showered on me at every stages of this work. We are greatly indebted to our Principal **Fr. Dr. Francis Karackat SDB**, Don Bosco Arts & Science College, Angadikadavu for providing the opportunity to take up this project as part of my curriculum.

We deeply indebted to our project guide **Mrs.Vineetha Mathew**, lectures of department of BCA, for her assistance and valuable suggestions as guide. She made this project a reality.

We express our sincere thanks to Mrs. Sindu PM, Mr. Hebin Layola, Mrs. Fincy Cyriac and Mrs. Sruthi N, lecturers of department of BCA, for their valuable suggestions during the course of this project. Their critical suggestions helped us to improve the project work.

Acknowledging the efforts of everyone, their chivalrous help in the course of the project preparation and their willingness to collaborate with the work, their magnanimity through lucid technical details lead to the successful completion of our project.

We would like to express our sincere thanks to all our friends, colleagues, parents and all those who have directly or indirectly assisted during this work.

# CONTENTS

CHAPTERS	CONTENTS	PAGE NO
1	INTRODUCTION	1
2	SYSTEM ANALYSIS	6
2.1	EXISTING SYSTEM	7
2.1.1	DISADVANTAGE OF EXISTING SYSTEM	7
2.2	PROPOSED SYSTEM	7
2.2.1	ADVANTAGE OF PROPOSED SYSTEM	8
2.3	FEASIBILITY STUDY	8
2.3.1	ECONOMIC FEASIBILITY	9
2.3.2	TECHNICAL FEASIBILITY	9
2.3.3	BEHAVIOURAL FEASIBILITY	9
2.4	SYSTEM SPECIFICATION	10
2.4.1	SOFTWARE SPECIFICATION	10
2.4.2	HARDWARE SPECIFICATION	10
2.5	IDENTIFICATION OF ACTORS	11
2.6	IDENTIFICATION OF USE CASES	11
2.6.1	USE CASES FOR THE ACTOR ADMINISTRATOR	12
2.6.2	USE CASES FOR THE ACTOR HOD	13
2.6.3	USE CASES FOR THE ACTOR STAFF	14
2.6.3	USE CASES FOR THE ACTOR STUDENT	14
2.6.4	USE CASE DIAGRAM	16
	SYSTEM DESIGN	20
3.1	INTRODUCTION	21



3.2	DATABASE DESIGN	21
3.3	TABLE DESIGN	22
3.4	AUTOMATED HELMET AND MASK VIOLATION DETECTION	23
3.5	DATA FLOW DIAGRAM	28
3.6	ER DIAGRAM	36
4	CODING	38
4.1	INPUT INTERFACE	39
4.2	OUTPUT INTERFACE	39
4.3	SOFTWARE DESCRIPTION	39
4.3.1	HTML	39
4.3.2	CSS	40
4.3.3	JAVASCRIPT	42
4.3.4	MYSQL	42
4.3.5	PYTHON	45
4.3.6	FLASK	46
5	CODING PAGES	47
5.1	ADMIN PAGE	48
5.2	HOD PAGE	49
5.3	STAFF PAGE	49
5.4	STUDENT	50
6	SYSTEM TESTING	51
6.1	TESTING AND EVALUATION	52
6.2	TESTING STRATEGIES	53
6.3	TESTING TECHNIQUES	54

6.3.1	WHITE BOX TESTING	54
6.3.2	BLACK BOX TESTING	54
6.3.3	UNIT TESTING	55
6.3.4	INTEGRATION TESTING	55
6.3.5	ACCEPTANCE TESTING	55
6.3.6	OUTPUT TESTING	56
7	SYSTEM IMPLEMENTATION AND DEPLOYMENT	59
8.1	CONCLUSION	60
8.2	REFERANCE	60
10	APPENDIX	61

# **CHAPTER I**

## **INTRODUCTION**

## **1.1 Project Overview**

Larva- Lab attendance register via application is a website which shifts our traditional methods of taking lab attendance into digital. Our project's aim is to digitally mark the lab attendance of students in our campus accordingly. The admin enters the details of all the HODs, staffs of various departments and manage the computer systems that are using in the lab. The HODs are assigned to enter the details of all the students including their phone number and photo. The students can register complaints regarding to the computer system that they are using. Notification goes to the admin profile regarding the complaint. All the attendance of the respective area can be viewed by the admin, HODs, staff and students. Teachers need to book the time slot in the lab for student. The approval is given by the admin.

## **NEED FOR THE SYSTEM**

Registering the attendance of students manually demands lots of space and effort and there is a chance for it to be missed out. For solving this problem, software is developed named LARVA (Lab Attendance Register via Application). This software also assists both teachers, in booking their time slots and students for registering grievances.

## **OBJECTIVES AND SCOPE**

Every tradition turns out to be great history with the times winged chariot. The field of computer applications is one great discipline of the globe that somersaults the face and pace of global performativity. Surprisingly, the computer labs of the discipline is still practicing the traditional attendance registers and it's time to practice it as part of the great legacy by introducing the newly designed software for attendance making i.e. LARVA which eases the process of collecting the attendance by matching the barcode of the student ID with the barcode scanner and later saved down for future reference.

The software also helps the students to occupy the systems by the scanner allotment if they miss at the actual allotted system based on the Roll number.

The process helps the faculties of all disciplines to collect the attendance of their students by logging in using their respective login and password. The software also helps the students to register complaints, if found any using LARVA.

The main objectives behind the development of this project are:

- 1) To mark lab attendance of the students.
- 2) To bring down the rate using paper register the campus.
- 3) To become a part of make digitalized campus.

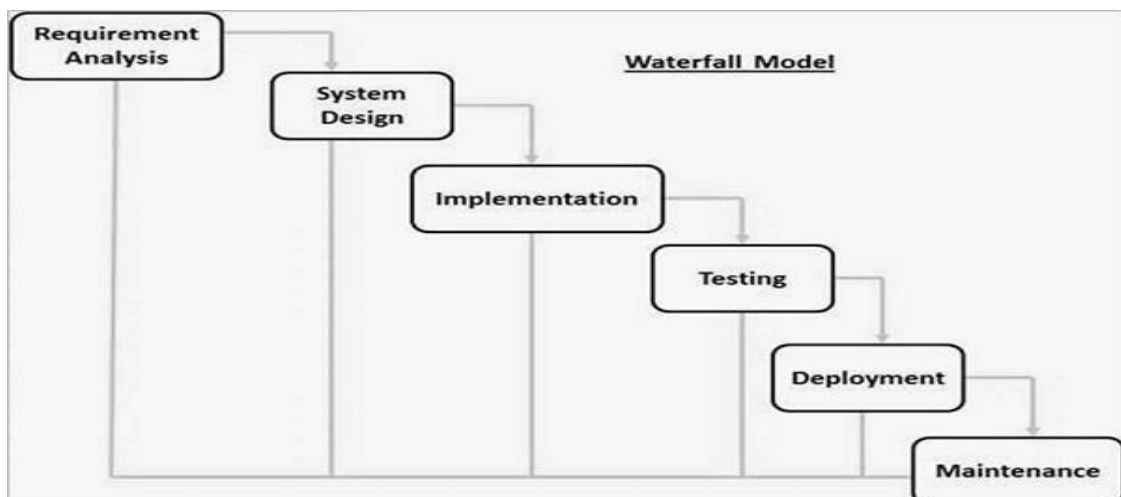
The scope of this project is high as it reduces manual work and utilises the resources in an efficient manner with minimum usage of time.

## **MODEL:**

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially

### **Waterfall Model - Design:**

In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. Following is the pictorial representation of Iterative and Incremental model:



The sequential phases in Waterfall model are –

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

## **Waterfall Model - Application:**

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are –

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short

## **The advantages of the waterfall model SDLC Model are as follows:**

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Some of the major advantages of the Waterfall Model are as follows –

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

**The disadvantages of the waterfall model SDLC Model are as follows:**

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows –

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang. At the very end, this doesn't allow identifying any technological or business bottleneck or challenges early.



# **CHAPTER II**

## **SYSTEM ANALYSIS**

## **INTRODUCTION**

System analysis is the process of collecting and interpreting facts, understanding problems and using the information to suggest improvement on the system. This will help to understand the existing system and determine how computers make its operation more effective. The aim of this analysis is to collect detailed information on the system and the feasibility study of the proposed system.

### **2.1 EXISTING SYSTEM**

Currently there are no digital methods for a computer lab attendance registration.

The traditional system of attendance registrations are done through the use of papers.

Attendance registrations through papers are time consuming process. Keeping the records of the attendance is space consuming.

#### **2.1.1 The Disadvantages of Existing System**

The existing system has the following disadvantages,

- There is no reliability for the paper data.
- Very difficult to manually check all attendance.

### **2.2 PROPOSED SYSTEM**

Since checking the attendance is a time consuming job for the teachers, we are trying to collect the lab attendance of the students, thereby making the teacher's job easier. This system can be established in college lab to collect attendance and teachers can book the time slot for the students. It will minimize paper dependency for collecting attendance.

In the proposed Lab Attendance Registration via Application, we are building a model to collect attendance of the students who are using the computer lab in the college. The system will collect the attendance through the barcode reader. For the attendance, the students need to scan the barcode in their ID card which is allocated by the college. The project will also help the students to register any complaints regarding to the computer

system they are using and is registered to the admin. The admin will rectify the complaint and then send reply to the student who registered the complaint.

LARVA will automatically allocate system for the student that meet the OS and Software specification of the selected subject. The admin can allocate software and OS for each computer systems through LARVA. After scanning the ID card the students can select the subject that they indeed to attend. The model will deny access to those students whose time slot has not been booked and approved by the teacher and the admin respectively.

### **2.2.1 Advantages of Proposed System**

- Automatic collection of attendance.
- Hassle free attendance system.
- Reduces paper usage.
- Time slot booking for efficient usage and distribution of computer lab time.

## **2.3 FEASIBILITY STUDY**

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioural feasibilities.

The proposed system is theoretically investigated to check the feasibility and found that they are more reliable and efficient in the cases given below. There are three aspects in the feasibility study portion of the preliminary investigation.

✓ Economic feasibility

✓ Technical feasibility

### ✓ Behavioural feasibility

The proposed system must be evaluated from a technical point of view first, and if technical feasible their impact on the organization must be assessed. If compatible, the operational system can be devised. Then they must be tested for economic feasibility.

#### **2.3.1 Economic Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors which affect the development of a new system is the cost it would require. Since the system developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

#### **2.3.2 Technical Feasibility**

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, procedures and staff. Having identified an outline system, the investigation must go on suggest the type of equipment, required method developing the system, of running the system once it has been designed. The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Though the technology become obsolete after some period of time, due to the fact that newer version of some software supports older versions, the system still be used. So there are only minimal constraints involved with this project. The system has been developed using C# and .NET, along with the database software SQL server, thus we could conclude that the project is technically feasible for development.

#### **2.3.3 Behavioural Feasibility**

People are inherently resistant to change and computers have been known to facilitate change. The System is designed in user friendly manner and we need to provide any special training for the persons using this software. The operating system used is Windows 10, which is also user friendly. Since the application is web biased and can easily access in a web browser, which is quite familiar to the intended users, it does not

have any operational barriers. So no need to provide any special training for using this application software and hence it is behaviourally feasible.

## **2.4 SYSTEM SPECIFICATIONS**

System Specification deals with the technical aspects the project has to meet in minimum to work successfully. This also includes the different aspects the software requirement is determined from. The technical details typically include:

- Software Specification
- Hardware Specification

### **2.4.1 Software Specifications**

The software required for the application depends on the following factors:

- ✓ The flexibility of the software
- ✓ Software contracts
- ✓ Limitation of the software

### **Software Requirement**

This specifies the minimum software requirements for implementing the system. This includes:

- Front End: HTML, CSS, Java Script, Python 3.9
- Back End: MySQL
- Client side scripting: java script
- Server side scripting: Python
- Platform: Flask
- Operating System: Microsoft windows 10

### **2.4.2 Hardware Specifications**

The software required for the application depends on the following factors:

- ✓ Determining size and capacity requirements.
- ✓ Computer evaluation and measurement.
- ✓ Financial factors.
- ✓ Maintenance and support.

## **Hardware Requirement**

- Microprocessor: Any 64 bit processor.
- Clock speed: - 2.13GHz
- Ram: 1 GB and above
- Hard disk: 40 GB and above
- Keyboard:-standard keyboard
- Mouse: Standard mouse
- Connectivity: - LAN & Wi-Fi
- Scanner: Standard Barcode scanner

## **2.5 IDENTIFICATION OF ACTORS**

A use cases represents the functionality of an actor. It is defined as a set of actions performed by a system, which yields an observable result. An ellipse containing its name inside the ellipse or below it represents. it. It is placed inside the system boundary and connected to an actor with an association. This shoes how the use cases and the actor interact.

We can identify the actors through a list of questions. The answers to these questions bring out the actors of the system is.

- Admin
- HOD
- Staff
- Student

Here we need to specify the use cases of each actor.

## **2.6 IDENTIFICATION OF USE CASES**

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yield an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

To find out the use cases, ask the following questions to each of the actors.

- ✓ Which functions does the actor require from the system? What does the actor need to do?
- ✓ Does the actor need to read, create, destroy, modify or store some kind of information in the system?
- ✓ Does the actor have to calculate something? And want to provide information for others?
- ✓ Could the actor's daily work be simplified or made more efficient by adding new functions to the system (typically functions which are currently not automated in the system)?

### **2.6.1 Use cases for the actor Administrator**

#### **Login:**

The first step involved is login. The admin can login to the website using username and password.

#### **Manage Department:**

Admin can add or remove departments.

#### **Manage Course:**

Admin can add or remove Courses of different department.

#### **Manage Staff:**

Admin can add or remove Staffs.

#### **Manage HOD:**

Admin can add or remove HOD.

#### **Manage System:**

Admin can add or remove Systems available.

#### **Manage software:**

Admin can add or remove software.

#### **Manage OS:**

Admin can add or remove OS.

**Manage Available Software:**

Admin can add or remove available software in each system.

**Manage Available OS:**

Admin can add or remove OS available in each system

**View timeslot and approval:**

Admin can view and approve lab time slot requested by the teachers.

**View Attendance:**

Admin can view and download attendance of all the students.

**Allocate subject to teachers:**

Admin will allocate subjects to each teacher.

**View Complaints:**

Admin can view student's complaints about the computer lab.

**Send reply:**

Admin can send replies to the registered complaints.

**2.6.2 Use cases for the actor HOD****Login:**

The HOD can login to the website using username and password.

**View Profile:**

HOD is able to view the profile.

**View Course:**

HOD can view the courses of the department.

**Subject Management:**

HOD can add or remove subjects of their department.

**View Teachers:**

HOD can view teachers of their department.



### **2.6.3 Use cases for the actor Staff**

#### **Login:**

Staff can login to the website using username and password.

#### **View profile:**

Staffs are able to view their profile.

#### **View allocated subject:**

Teachers can view their allocated subjects.

#### **Request for timeslot:**

Teachers can book timeslot in lab of their allocated subjects for the students.

#### **View request status:**

Teachers can view timeslot request status.

#### **Add student remark:**

Teachers can add remarks of the student.

#### **View attendance and download:**

Teachers can view and download subject wise student attendance.

### **2.6.4 Use case for the actor student**

#### **Login:**

Students can login to the website using username and password.

#### **View profile:**

Students are able to view their profile.

#### **View subject:**

Students can view their subjects for the lab.

#### **View timeslot:**

View the timeslot for the lab with respected subject that are booked by the teacher.

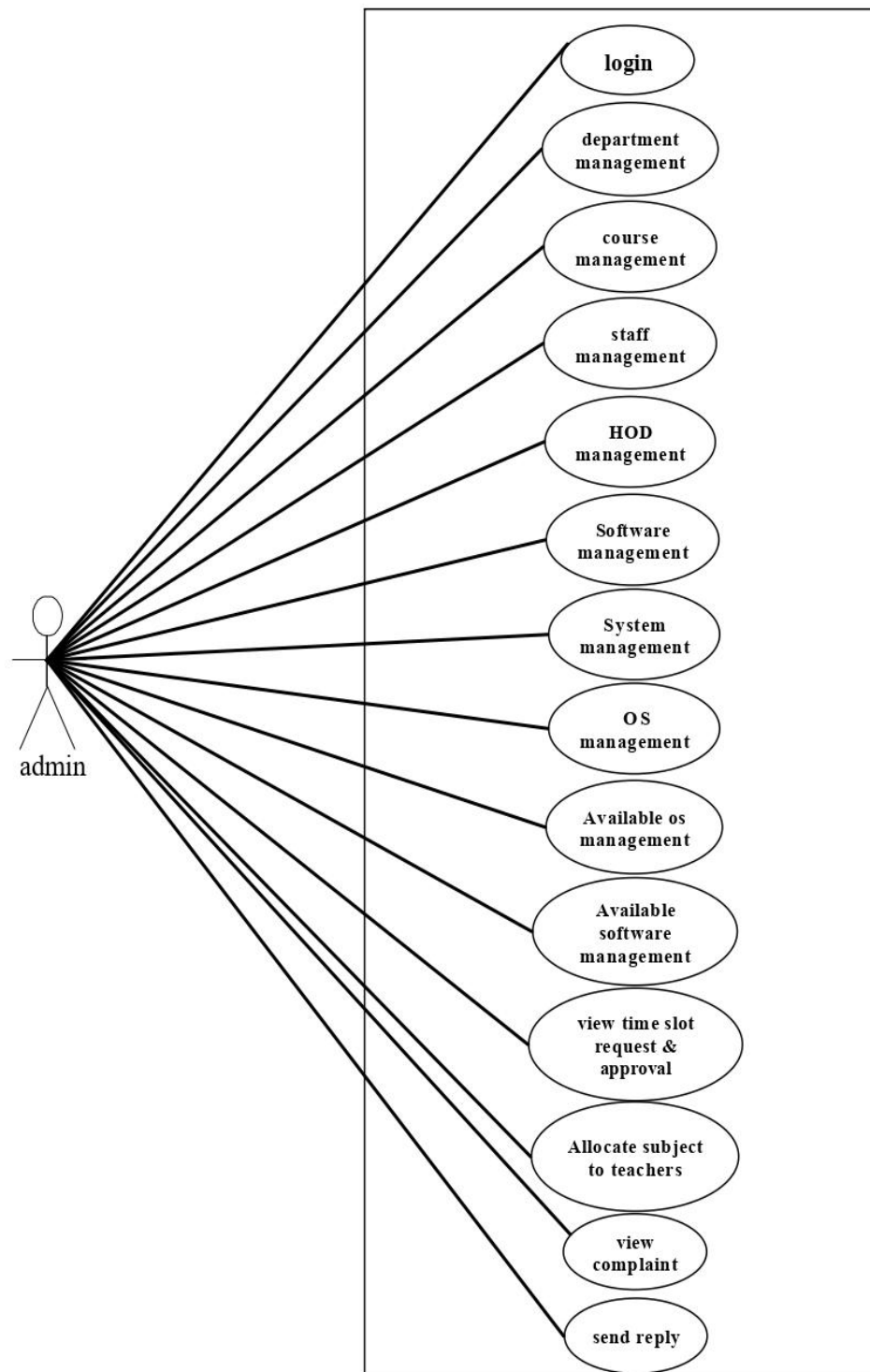
**Send complaints:**

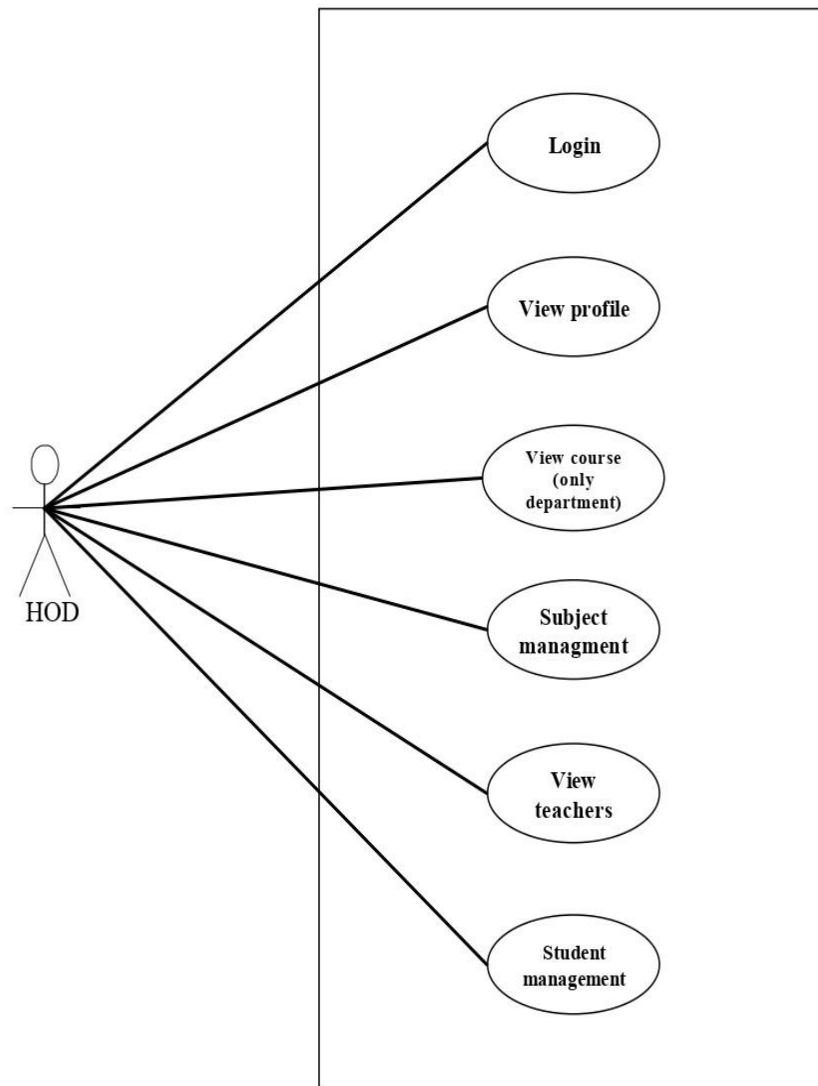
Students can send complaints to the admin regarding the computer system that they are using.

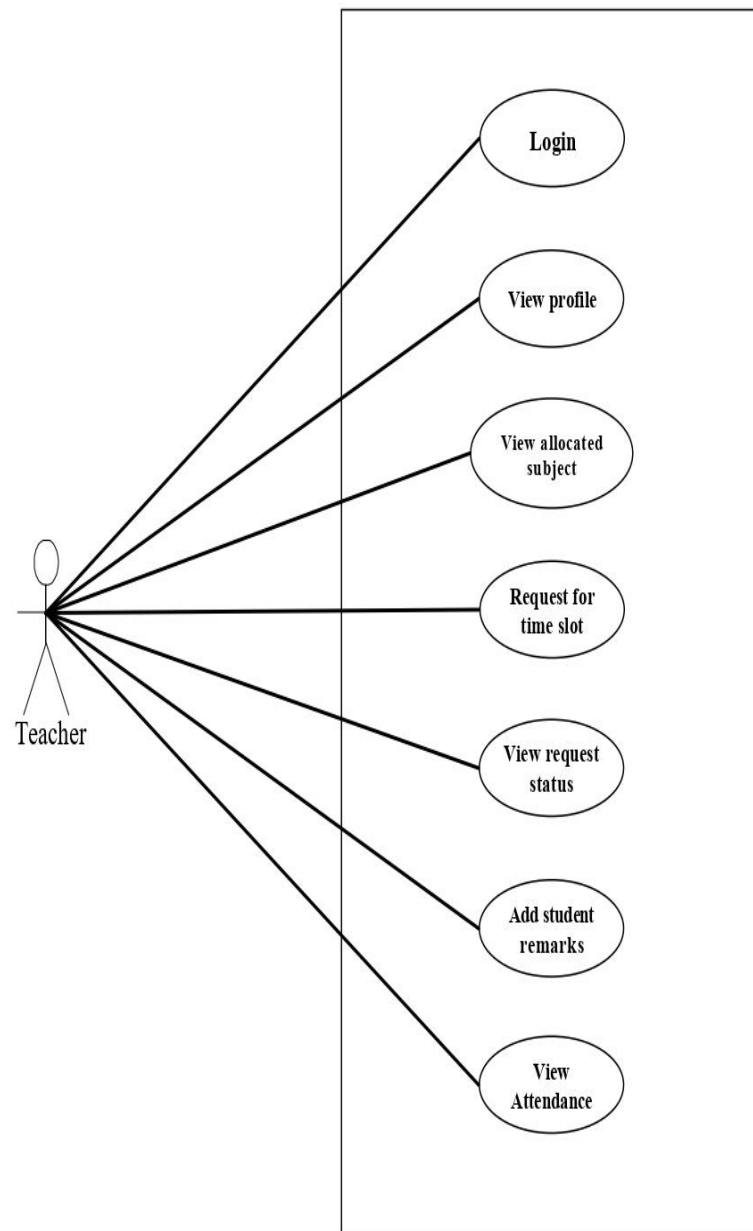
**View reply:**

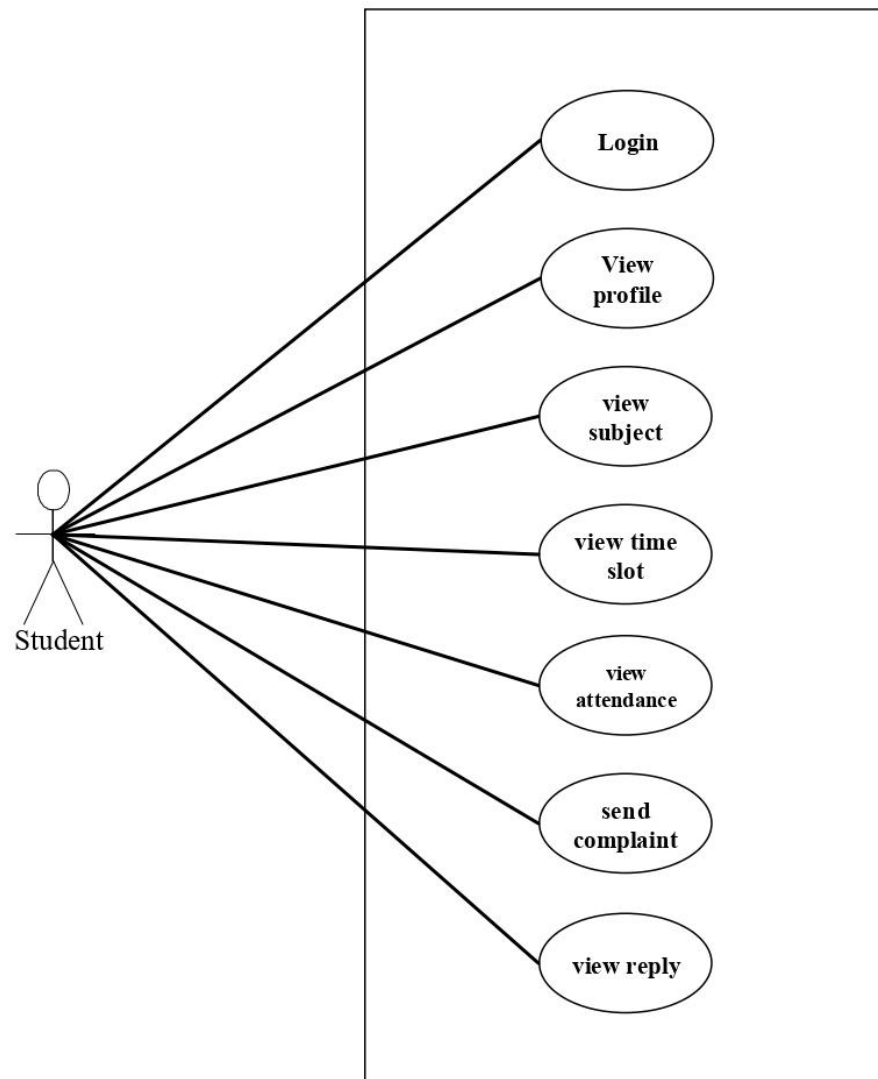
Students can view the reply of their registered complaints.

## 2.6.7 USE CASE DIAGRAM









# **CHAPTER III**

## **SYSTEM DESIGN**

### **3.1 INTRODUCTION**

System design provides an understanding of the procedural details, necessary implementing the system recommended in the feasibility study. Basically it is all about the creation of a new system. This is a critical phase since it decides the quality of the system and has a major impact on the testing and implementation phases.

**System design consists of three major steps.**

- Drawing of the expanded system data flow charts to identify all the processing functions required.
- The allocation of the equipment and the software to be used.
- The identification of the test requirements for the system.

### **CHARACTERS OF DESIGN**

- A design should exhibit a hierarchical organization that makes intelligent use of control among components of the software.
- A design should be modular that is, the software should be logical.
- A design should contain distinct and separable representation of data and procedure.
- A design should lead to interface that reduce the complexity of the connections between modules and with the external environment.

### **3.2 Database Design**

A Database is a collection of inter related data stored with minimum redundancy to serve many users quickly and efficiently. In database design data independence, accuracy, privacy and security are given higher priority. Database design is an integrated approach to the file design. This activity deals with the design of the physical data base. All entities and attributes have been identified while creating the database. The database design deals with the grouping of data into number of tables so as to.

- ✓ Reduplication of data.
- ✓ Minimize storage space.
- ✓ Retrieve the data efficiently.



Following are some guidelines for the database design:

- Design a relational schema so that it is easy to explain its meaning. Do not combine attributes from multiple entry and relationship type into a single relation.
- Design the database schema so that no insertion, deletion or modification anomalies are present in the relation.
- As far as possible, avoid placing attributes in the base relation whose values may frequently be null.
- Design relation schema so that they can be joined with equality conditions on attributes that are either primary keys or foreign keys in a way that no spurious tuples are generated.

### **3.3 Table Design**

DB design is required to manage large bodies of information. The management of data involves both the definition of the structure of storage of information and provisions of mechanism for the manipulation of information. For developing an efficient database certain conditions have to be fulfilled such as:

- Control Redundancy
- Ease of Use
- Data Independence
- Accuracy and Integrity

There are five major steps in design process:

- Identify the table and relationship.
- Identify the data that is needed for each table and relationship.
- Resolve the relationship.
- Verify the design.
- Implement the design

The Database Consist of the following tables given below.

### 3.4 LARVA-Lab Attendance Registration via Application

#### 1. Login table

Colum name	Data type	Constraints	Description
l_id	int	primary key	unique identifier
user_name	varchar(50)	not null	name of user
password	varchar(50)	not null	secret key
usertype	varchar(50)	not null	To specify the role

#### 2. Allocate\_subject table

Colum name	Data type	Constraints	Description
allocate_id	int	primary key	unique identifier
sub_id	int	not null	
staff_id	int	not null	

#### 3. Allocation\_system table

Colum name	Data type	Constraints	Description
alloc_sys_id	Int	primary key	unique identifier
time_slot_id	int	not null	
sys_id	int	not null	
stud_id	int	Not null	

#### 4. Attendance table

Colum name	Data type	Constraints	Description
attendance_id	int	primary key	unique identifier
time_slot_id	int		time slot id
student_id	int		Student id
attendance	varchar(20)	not null	

### 5. Available OS table

Colum name	Data type	Constraints	Description
available_id	int	primary key	unique identifier
sys_id	int	foreign key	system id
os_id	int	not null	os id

### 6. Available software table

Colum name	Data type	Constraints	Description
available_id	int	primary key	unique identifier
sys_id	int	not null	system id
software_id	int	not null	software id

### 7. Complaint table

Colum name	Data type	Constraints	Description
complaint_id	int	foreign key	login identifier
date	varchar(20)	not null	date
student_id	int	not null	
complaint	varchar(100)	not null	complaint
reply	varchar(100)	not null	reply
reply_date	varchar(100)		reply date

### 8. Course table

Colum name	Data type	Constraints	Description
course_id	int	foreign key	login identifier
dep_id	int	not null	department id
course_name	varchar(20)	null	course name

### 9. Department table

Colum name	Data type	Constraints	Description
dep_id	int	Not null	login identifier
dep_name	varchar(20)	not null	department name

### 10. HOD table

Colum name	Data type	Constraints	Description
hod_id	int	Primary key	login identifier
staff_id	int	not null	staff id

### 11. OS table

Column name	Data type	Constraints	Description
os_id	int	Primary key	login identifier
os_name	varchar(20)	not null	os name
version	varchar(20)	not null	version

### 12. Remark table

Column name	Data type	Constraints	Description
remark_id	int	Primary key	login identifier
student_name	int	not null	student_id
remark	varchar(500)	not null	Reamrk
date	date		date

### 13. Software table

Column name	Data type	Constraints	Description
software_id	int	Primary key	login identifier
software_name	int	not null	software name
description	varchar(100)	not null	description

#### 14. Staff table

Colum name	Data type	Constraints	Description
staff_id	int	Primary key	login identifier
dep_id	int	not null	department id
name	varchar(20)	not null	name
email	varchar(20)	not null	email
phone	bigint	not null	phone number
qualification	varchar(20)	not null	qualification
house	varchar(20)	not null	house name
place	varchar(20)	not null	place
post	varchar(20)	not null	post
image	varchar(200)		image

#### 15. Student table

Colum name	Data type	Constraints	Description
student_id	int	Primary key	login identifier
name	varchar(20)	not null	name
course_id	int	not null	course id
semester	varchar(20)	not null	semester
register_no	varchar(20)	not null	register number
date_of_birth	varchar(20)	not null	date of birth
phone	bigint	not null	phone number
palce	varchar(20)	not null	palce
post	varchar(20)	not null	post
image	varchar(500)	not null	image
admission_no	varchar(20)	not null	admission number

### 16. Subject table

Colum name	Data type	Constraints	Description
sub_id	int	foreign key	login identifier
course_id	int	not null	course_id
sem	varchar(20)	not null	semester
complaint	varchar(100)	not null	complaint
subject_name	varchar(30)	not null	subject name
software_id	int	not null	software id
os_id	int	not null	os id

### 17. System table

Column name	Data type	Constraints	Description
sys_id	int	Primary key	login identifier
sys_no	int	not null	system number
mac_address	varchar(50)	not null	mac address

### 18. Time slot table

Colum name	Data type	Constraints	Description
slot_id	int	foreign key	login identifier
sub_id	int	not null	subject id
date	varchar(20)	not null	date
time_from	varchar(20)	not null	time from
time_to	varchar(20)	not null	time to
status	varchar(30)	not null	status

### **3.5. DATA FLOW DIAGRAM**

A graphical representation is used to describe and analyses the movement of data through a system manual or automated including the processes, Storing of data and delays in the system. Data flow diagrams are the central tool and the basis from which other components are developed.

The transformation of data, from input to output through process may be described logically and independently of the physical components associated with the system.

They are termed logical dataflow diagrams, showing the actual implementations and the movement of data between people, departments and workstations. DFD is one of the most important modelling tools used in system design. DFD shows the flow of data through different process in the system.

#### **PURPOSE:**

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have reasonably completed model of the behaviour of the system. It is important to avoid premature designs, wherein develop designs before analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

Throughout my project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In my opinion, "efficient design of the data flow and context flow diagram helps to design the system successfully without much major flaws within the scheduled time". This is the most complicated part in a project. In the designing process, my project took more than the activities in the software lifecycle. If we design a system efficiently with all the future enhancements the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as a way of expressing system requirements in graphical form. A data flow diagram also known as "bubble chare" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It functionality

decompose sees the requirement specification down to the lowest level. Data Flow Diagram depicts the information flow, the transformation flow and the transformations that are applied as data move from input to output. Thus DFD describes what data flows rather than how they are processed.

Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is one of the most important tools used during system analysis. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

Data Flow Diagrams are made up of a number of symbols, which represents system components. Data flow modelling method uses four kinds of symbols, which are used to represent four kinds of system components.

These are

- Process
- Data stores
- Data flows
- External entity

#### **Process:**

Process shows the work of the system. Each process has one or more data inputs and produce one or more data outputs. Processes are represented by rounded rectangles in Data Flow Diagram. Each process has a unique name and number. This name and number appears inside the rectangle that represents the process in a Data Flow Diagram.

#### **Data Stores:**

A data stores is a repository of data. Processes can enter data, into a store or retrieve the data from the data store. Each data has a unique name.

#### **Data Flows:**

Data flows show the passage of data in the system and are represented by lines joining system components. An arrow indicates the direction of flow and the line is labelled by name of the dataflow.



**External Entity:**

External entities are outside the system but they either supply input data into the system or use other systems output. They are entities on which the designer has control. They may be an organizations customer or other bodies with which the system interacts. External entities that supply data into the system are sometimes called source. External entities that use the system data are sometimes called sinks. These are represented by rectangles in the Data Flow Diagram.

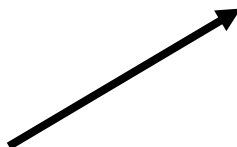
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Basic data flow diagram symbols are.....

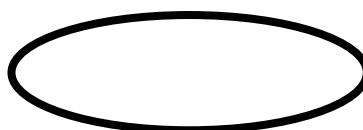
- A Square defines a source (originator) or destination of a system data:



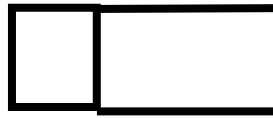
- An Arrow identifies data flow. It is a pipeline through which information flows:



- A Circle represents a process that transforms incoming data flow(s) into outgoing data flow(s):



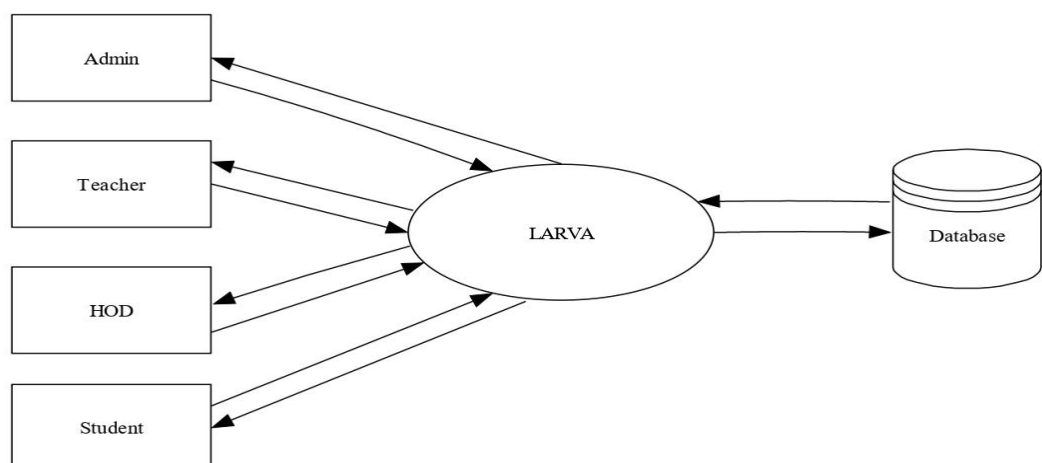
- An Open Rectangle is a data store:



#### Four steps are commonly used to construct a DFD:

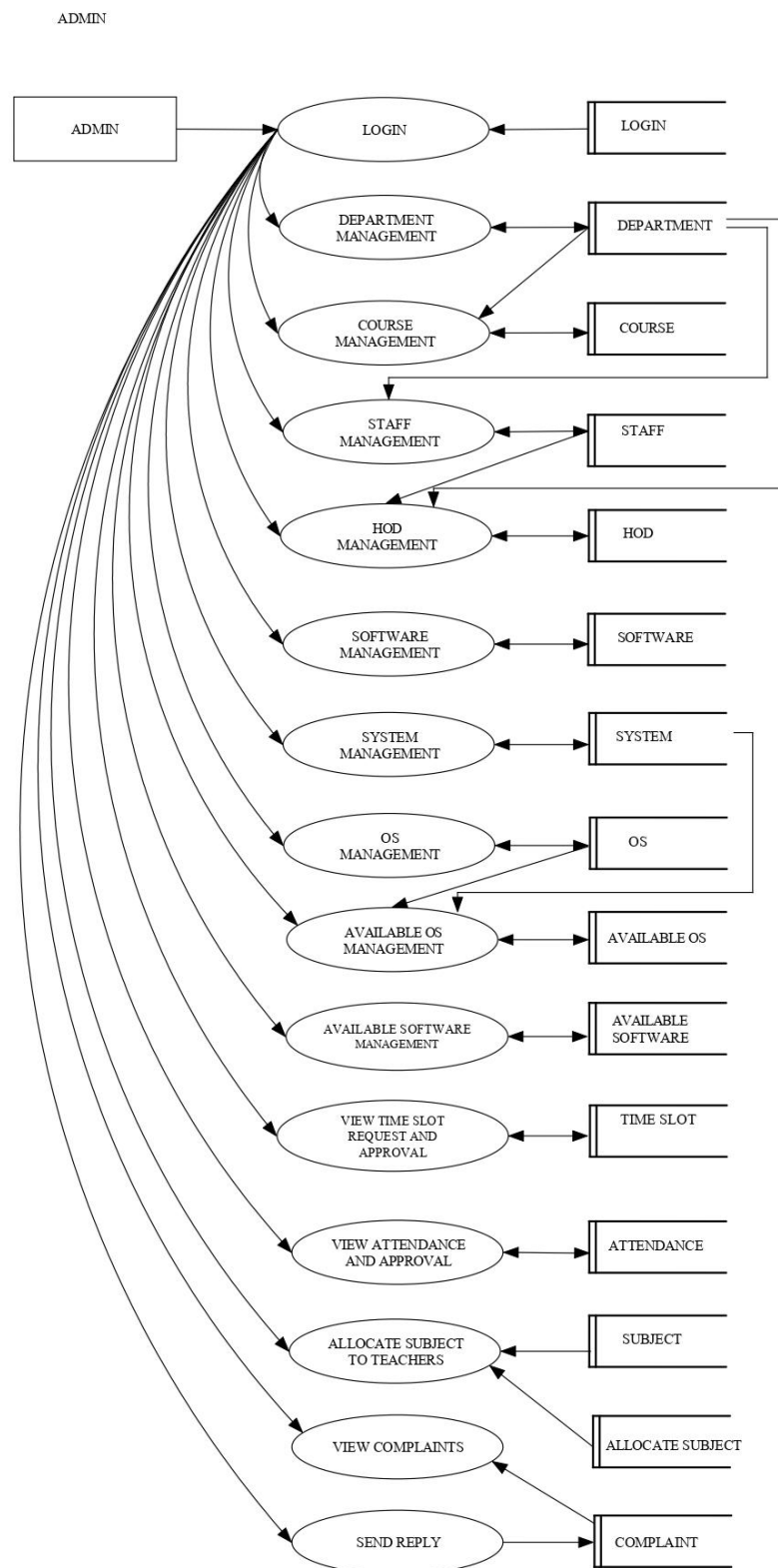
- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and left to right.
- When a process is exploded in to lower level details they are numbered.
- The names of data stores, sources and destinations are written in Capital letters.

#### DFD Level-0



## DFD Level-1.1

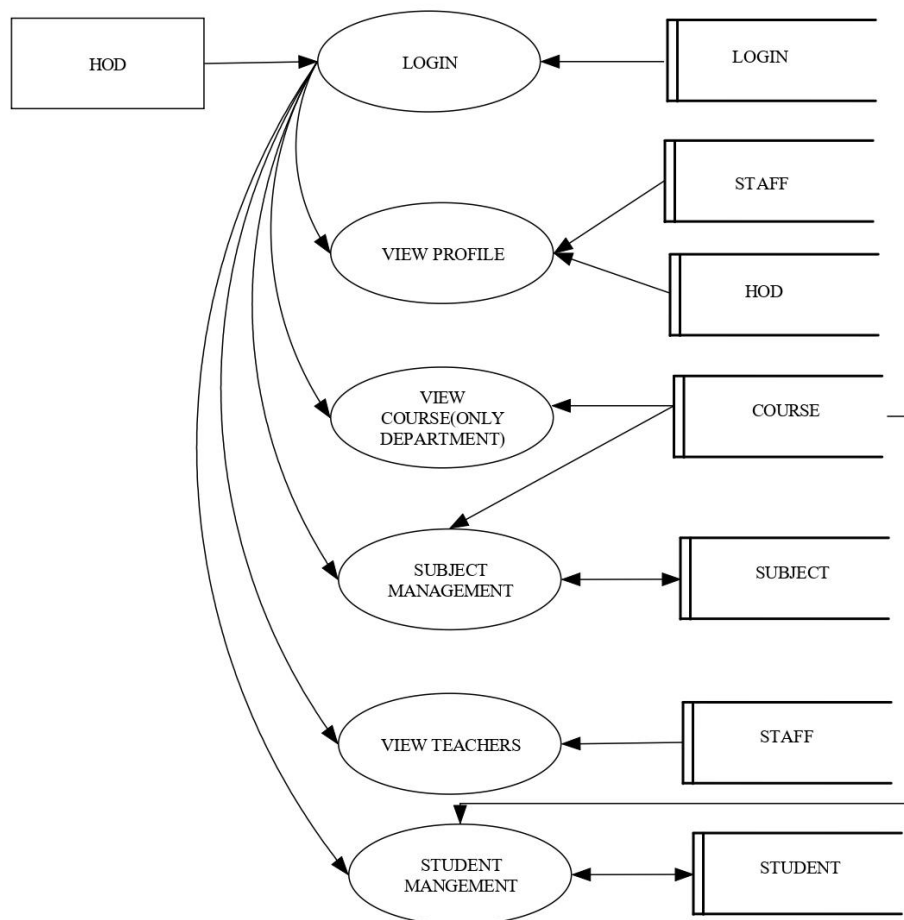
### Admin



## DFD Level-1.2

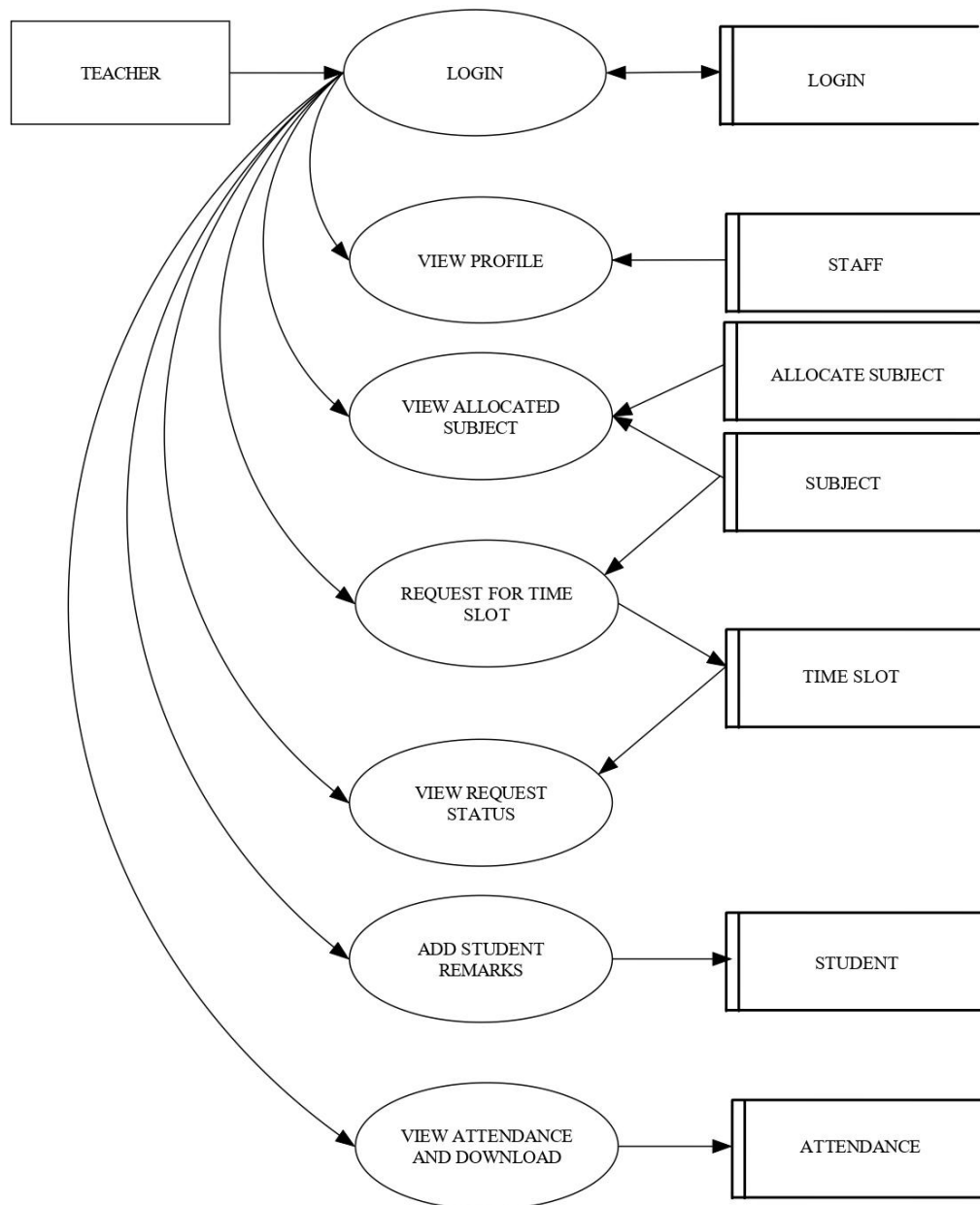
### HOD

HOD



## DFD Level-1.3

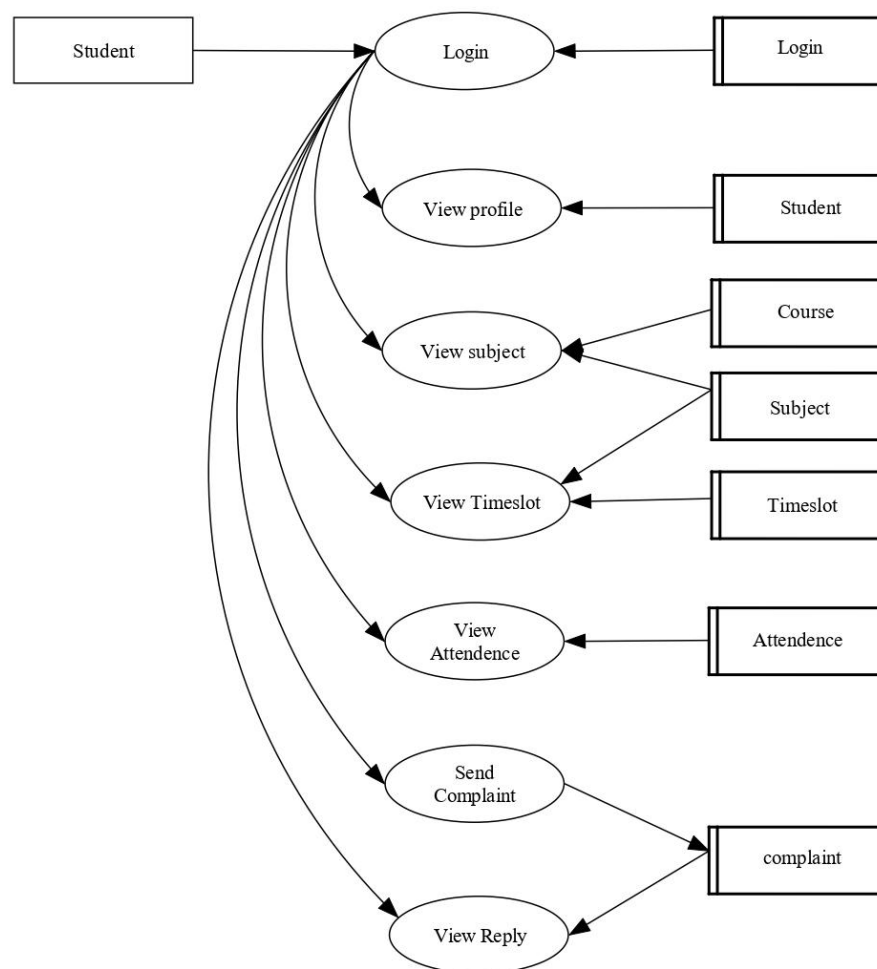
### Staff



## DFD Level-1.4

### Student

STUDENT



### 3.6 ER Diagram

An ER diagram is a diagram that helps to design databases in an efficient way. It is a data model for describing the data or information. It is a visual representation of data that describes how data is related to each other. The main components of ER models are entities, attributes and the relationships that can exist among them.

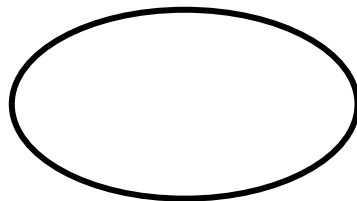
#### Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.



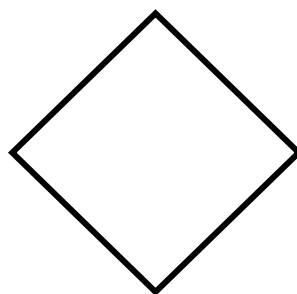
#### Attribute

Attributes are properties of entities. Attributes are represented by means of eclipses. Every eclipse represents one attribute and is directly connected to its entity (rectangle).

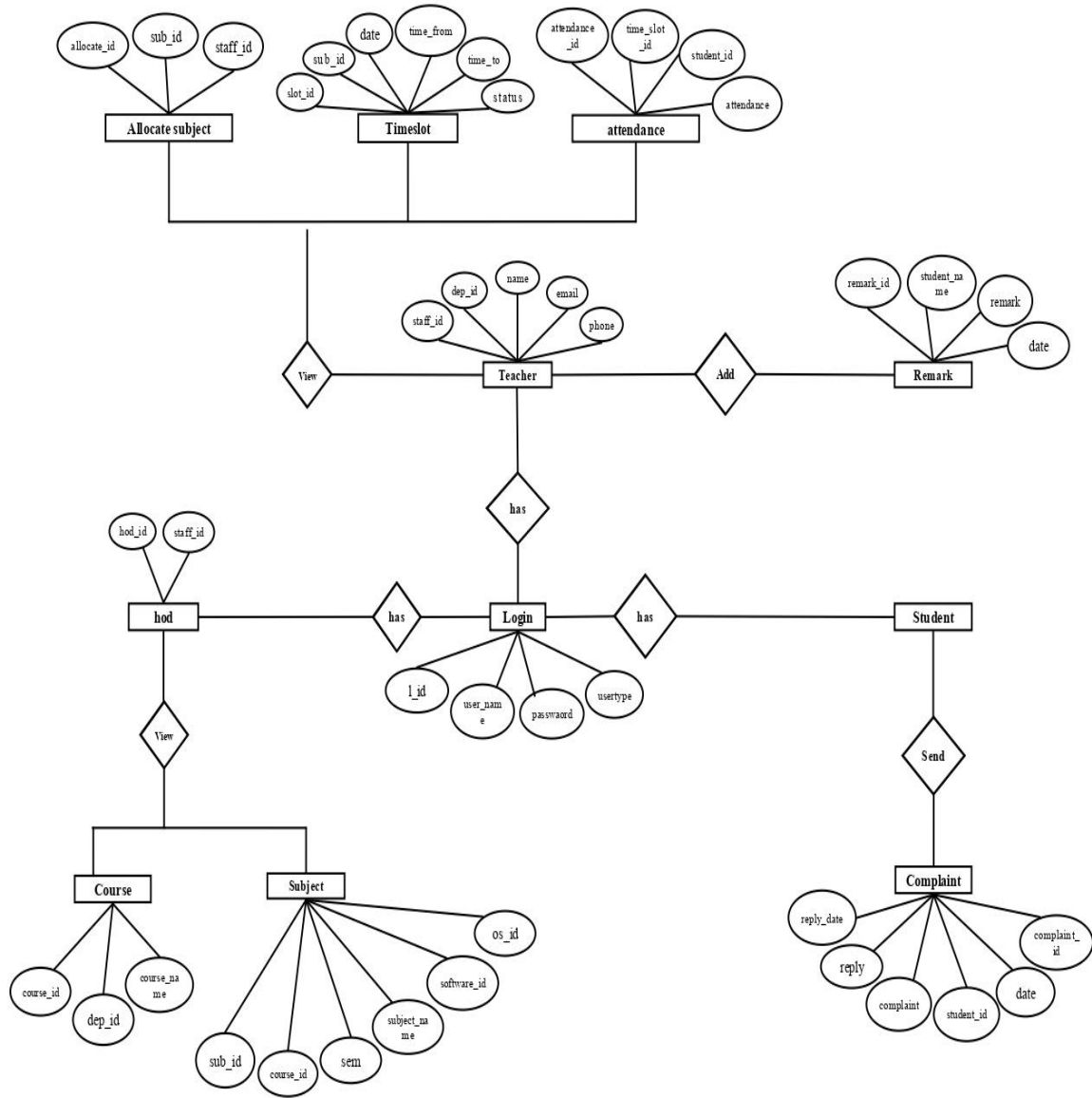


#### Relationship

Relationships are represented by diamond shaped box. Name of the relationship is written in the diamond box. All entities (rectangles), participating in relationship, are connected to it by a line.



## Architectural design





# **CHAPTER IV**

## **CODING**

## **4.1 INPUT INTERFACE**

Input design is a part of overall system design, which requires very careful attention. If data going into the system is correct, then the processing and output will magnify these errors. Thus the designer has a number of clear objectives in the different stages of input design.

- To produce a cost effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that input is acceptable to and understand by the user.

## **4.2 OUTPUT INTERFACE**

At the beginning of the output design various types of outputs such as external, internal, operational and interactive and turn around are defined. Then the format, content, location, frequency, volume and sequence of the outputs are specified. The content of the output must be defined in detail. The system analysis has two specific objectives at this stage.

- To interpret and communicate the results of the computer part of a system to the users in a form, which they can understand, and which meets their requirements.
- To communicate the output design specifications to programmers in a way in which it is unambiguous, comprehensive and capable of being translated into a programming language.

## **4.3 SOFTWARE DESCRIPTION**

### **4.3.1 HTML**

Hypertext Mark-up Language (HTML) is the standard mark-up language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML

specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

HTML files are written in ASCII text, so the user can use any text editor to create his/her web page, though a browser of one sort or another is necessary to view the web page. HTML is case insensitive with its language commands. The characters within the document, however, are case sensitive. The language consists of various "tags" which are known as elements. These allow the browser to understand (and put into the desired/specified format) the layout, background, headings, titles, lists, text and/or graphics on the page. The elements are classified according to their function in the HTML document. There are head elements and body elements. The head elements identify properties of the entire document, while body elements actually mark text as content and show a change in the appearance in one way or another. Most elements have a beginning and an ending which encompass the text the user wishes to mark with the tag. All HTML documents must begin with the element and end with the element. Some of the other elements which may be used are tags to create lists-- both ordered lists as well as unordered lists. The user may also create larger or smaller, bolder, italicized, or underlined text. Attributes may be used along with the elements. These perform functions such as placement of text, indication of the source files of images, and identification of links to the document or part of the document.

#### **4.3.2 CSS**

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a mark-up language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages, user interfaces for web applications, and user interfaces for many mobile applications. CSS is designed primarily to enable the separation of document

content from document presentation, including aspects such as the layout, colours, and fonts.

### **Advantages of CSS**

- CSS saves time – you can write CSS once and then reuse same sheet in multiple HTML pages.

You can define a style for each HTML element and apply it to as many Web pages as you want.

- Pages load faster – If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
- Easy maintenance – To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- Superior styles to HTML – CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- Multiple Device Compatibility – Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- Global web standards – Now HTML attributes are being deprecated and it is being recommended to use CSS. So it is a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.
- Offline Browsing – CSS can store web applications locally with the help of an offline cache.  
Using of this, we can view offline websites. The cache also ensures faster loading and better overall performance of the website.
- Platform Independence – The Script offer consistent platform independence and can support latest browsers as well.

### **4.3.3 JAVASCRIPT**

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as Live Script, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name Live Script.

The General-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

#### **Advantages of JavaScript:**

- Less server interaction – you can validate user input before sending the page off to the server. This saves server traffic, which means fewer loads on your server.
- Immediate feedback to the visitors – they don't have to wait for a page reload to see if they have forgotten to enter something.
- Increased interactivity – you can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.
- Richer interfaces – you can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

### **4.3.4 MySQL**

MySQL is an open-source relational database management system (RDBMS). MySQL is released under an open-source license. So you have nothing to pay to use it. MySQL is a very powerful program in its own right.

It handles a large subset of the functionality of the most expensive and powerful database packages. It uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.

It works very quickly and works well even with large data sets. It is very friendly to PHP, the most appreciated language for web development. It supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit

of 8 million terabytes (TB). It is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

### **Major features as available in MySQL 5.6**

- A broad subset of ANSI SQL 99, as well as extensions.
- Cross-platform support.
- Stored procedures, using a procedural language that closely adheres to SQL/PSM.
- Triggers.
- Cursors.
- Updatable views.
- Online DDL when using the InnoDB Storage Engine.
- Information schema.
- Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
- A set of SQL Mode options to control runtime behaviour, including a strict mode to better adhere to SQL standards.
- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine.
- Transactions with savepoints when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
- ACID compliance when using InnoDB and NDB Cluster Storage Engines.
- SSL support → Query caching → Sub-SELECTs (i.e. nested SELECTs) .
- Built-in replication support (i.e., master-master replication and master-slave replication) with one master per slave, many slaves per master.
- Multi-master replication is provided in MySQL Cluster, and multimaster support can be added to unclustered configurations using Galera Cluster.
- Full-text indexing and searching.
- Embedded database library.
- Unicode support.
- Partitioned tables with pruning of partitions in optimizer.
- Shared-nothing clustering through MySQL Cluster.
- Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.

- Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.
- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

## **Advantages**

MySQL database server has lots of advantages over its competitors. Some of these advantages have been explained below.

- **Open Source and Cost Effective:**

The best thing about MySQL server is that this is open source and it has a free version as well.

By open source software, we mean that the code of the software is available and anyone can tailor it according to his requirement. Companies prefer MySQL because they don't have to pay anything for this excellent product.

- **Portability:**

MySQL is cross platform database server. MySQL can be run on a variety of platforms including Windows, OS2, Linux and Solaris. Portability of MySQL server makes it suitable for applications that target multiple platforms particularly web application. MySQL contains API for almost all the major programming languages and can be easily integrated with the languages like PHP, C++, Perl, C, Python and ruby. In fact, MySQL is a part of the famous LAMP (Linux Apache MySQL PHP) server stack which is used worldwide for web application development.

- **Seamless Connectivity:**

Various secure and seamless connection mechanisms are available in order to connect with MySQL server. These connections include named pipes, TCP/IP sockets and UNIX Sockets.

- **Rapid Development and Continuous Updates:**

Being an open source product, MySQL has a very large developer community which releases regular patches and updates for MySQL. Several database templates have been developed which can be readily used and modified resulting in rapid application development.

➤ **Security:**

MySQL server databases are extremely secure and all the data access scenarios are protected via password and good thing about these passwords is that they are stored in encrypted form and it is not easy to break these advanced and complex encryption algorithms.

#### **4.3.5 Python**

Python is an interpreter, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding; make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

#### **Major features of python**

- Easy to code
- Free and Open Source
- Object-Oriented Language



- GUI Programming Support
- High-Level Language
- Extensible feature
- Python is Portable language
- Python is Integrated language

### **Advantages**

- Extensive support libraries
- Integration feature
- Improved productivity
- Easy to learn and write
- Vast library support
- Free and open source

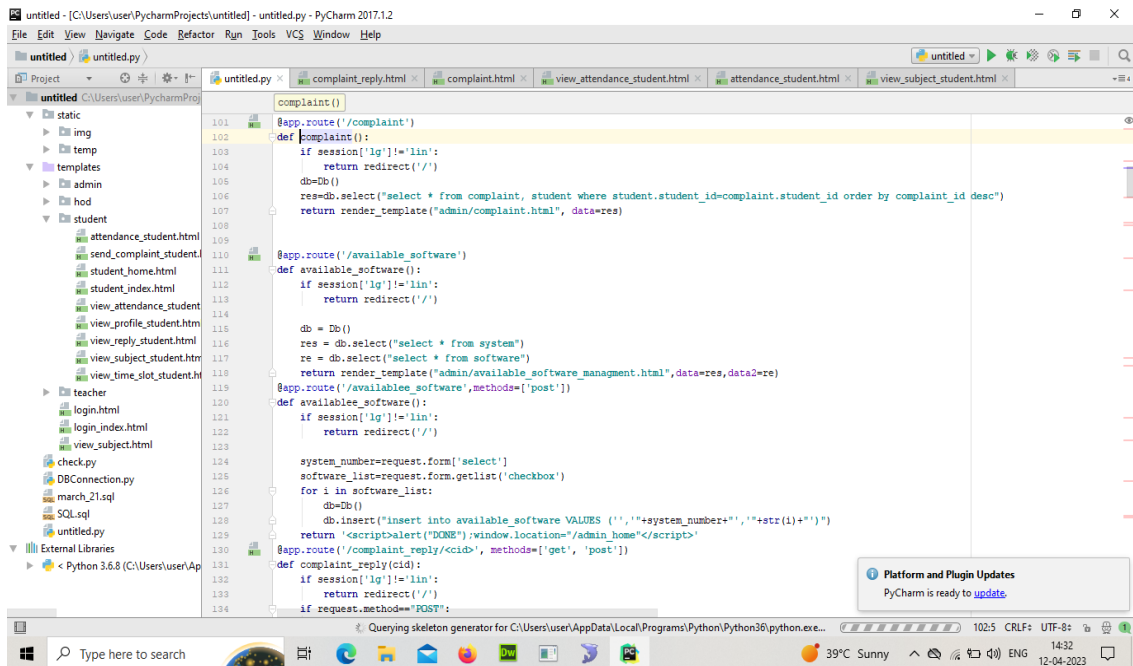
### **4.3.6 Flask**

Flask is a web application framework written in Python. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Pocco. Flask is based on the Werkzeug WSGI toolkit and the Jinja2 template engine. Both are Pocco projects.

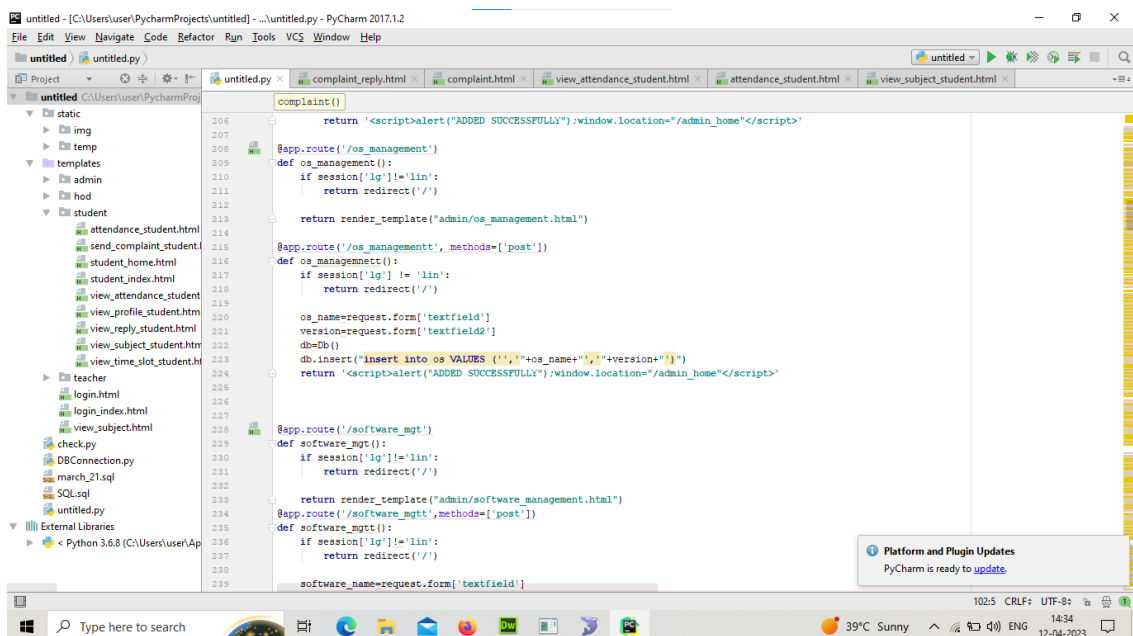
It is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier. It gives developers flexibility and is a more accessible framework for new developers since you can build a web application quickly using only a single Python file. Flask is also extensible and doesn't force a particular directory structure or require complicated boilerplate code before getting started.

**CHAPTER V**  
**CODING PAGES**

## 5.1 Admin Page

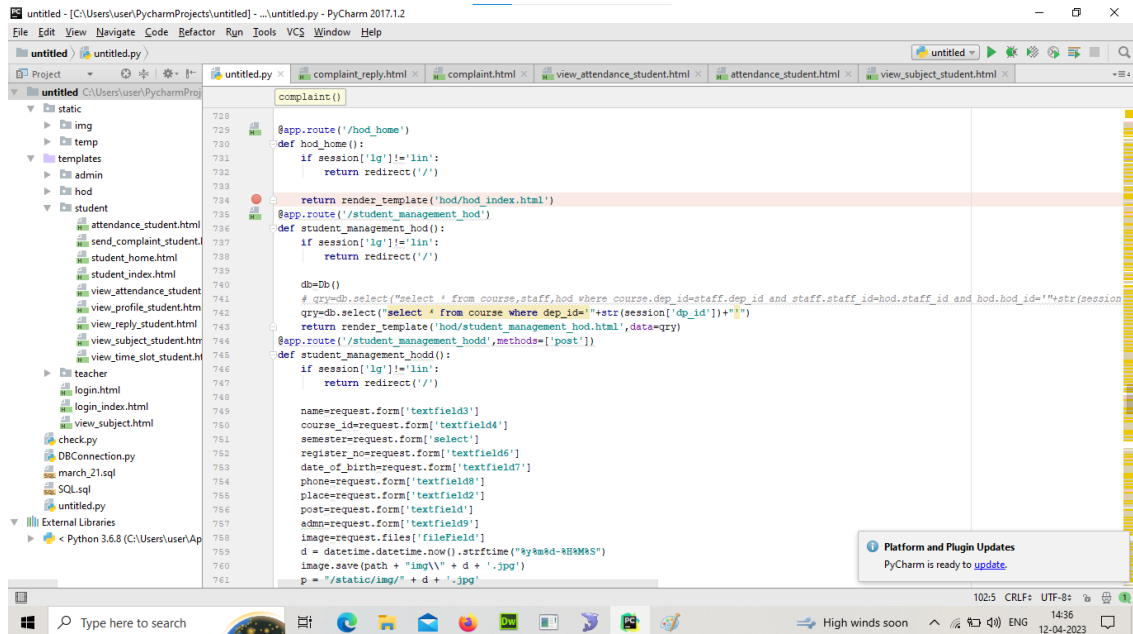


```
101 def complaint():
102     @app.route('/complaint')
103     def complaint():
104         if session['lg']!='lin':
105             return redirect('/')
106         db=Db()
107         res=db.select("select * from complaint, student where student.student_id=complaint.student_id order by complaint_id desc")
108         return render_template("admin/complaint.html", data=res)
109
110 @app.route('/available_software')
111 def available_software():
112     if session['lg']!='lin':
113         return redirect('/')
114
115     db = Db()
116     res = db.select("select * from software")
117     re = db.select("select * from software")
118     return render_template("admin/available_software_management.html", data=res, data2=re)
119
120 @app.route('/available_software', methods=['post'])
121 def available_software():
122     if session['lg']!='lin':
123         return redirect('/')
124
125     system_number=request.form['select']
126     software_list=request.form.getlist('checkbox')
127     for i in software_list:
128         db=Db()
129         db.insert("insert into available_software VALUES ('"+system_number+"','"+str(i)+"")")
130     return <script>alert("DONE");window.location="/admin_home"</script>
131
132 @app.route('/complaint_reply/cid', methods=['get', 'post'])
133 def complaint_reply(cid):
134     if session['lg']!='lin':
135         return redirect('/')
136     if request.method=="POST":
137         ...
```

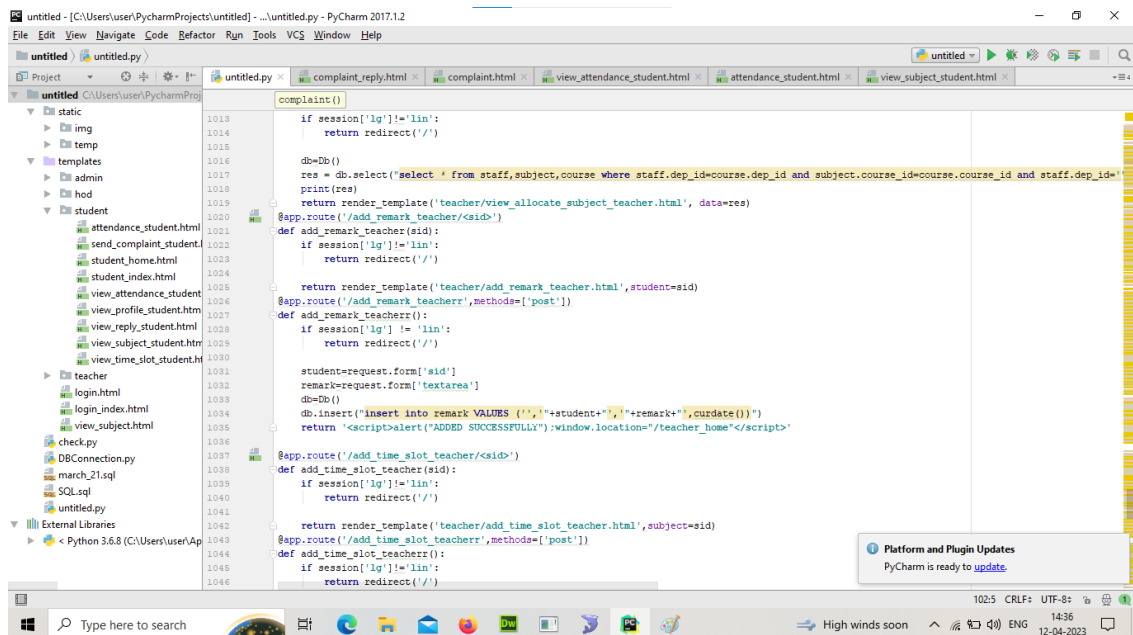


```
206 return <script>alert("ADDED SUCCESSFULLY");window.location="/admin_home"</script>
207
208 @app.route('/os_management')
209 def os_management():
210     if session['lg']!='lin':
211         return redirect('/')
212
213     return render_template("admin/os_management.html")
214
215 @app.route('/os_management', methods=['post'])
216 def os_management():
217     if session['lg']!='lin':
218         return redirect('/')
219
220     os_name=request.form['textfield']
221     version=request.form['textfield2']
222     db=Db()
223     db.insert("insert into os VALUES ('"+os_name+"','"+version+"")")
224     return <script>alert("ADDED SUCCESSFULLY");window.location="/admin_home"</script>
225
226 @app.route('/software_mgt')
227 def software_mgt():
228     if session['lg']!='lin':
229         return redirect('/')
230
231     return render_template("admin/software_management.html")
232
233 @app.route('/software_mgt', methods=['post'])
234 def software_mgt():
235     if session['lg']!='lin':
236         return redirect('/')
237
238     software_name=request.form['textfield']
239     ...
```

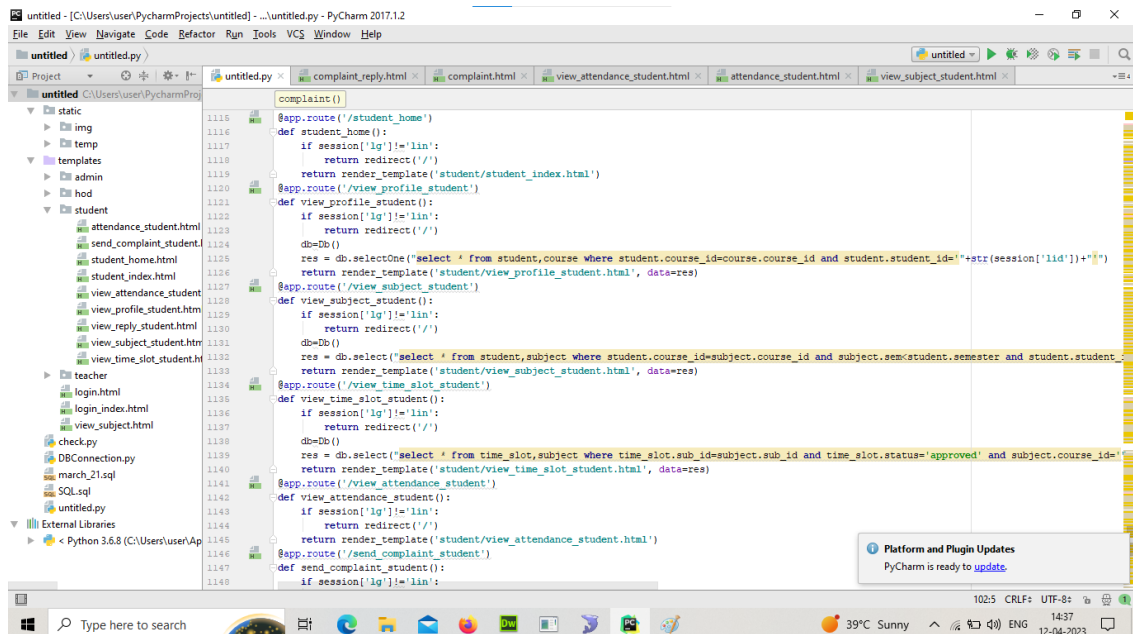
## 5.2 HOD PAGE



### 5.3 STAFF



## 5.4 STUDENT

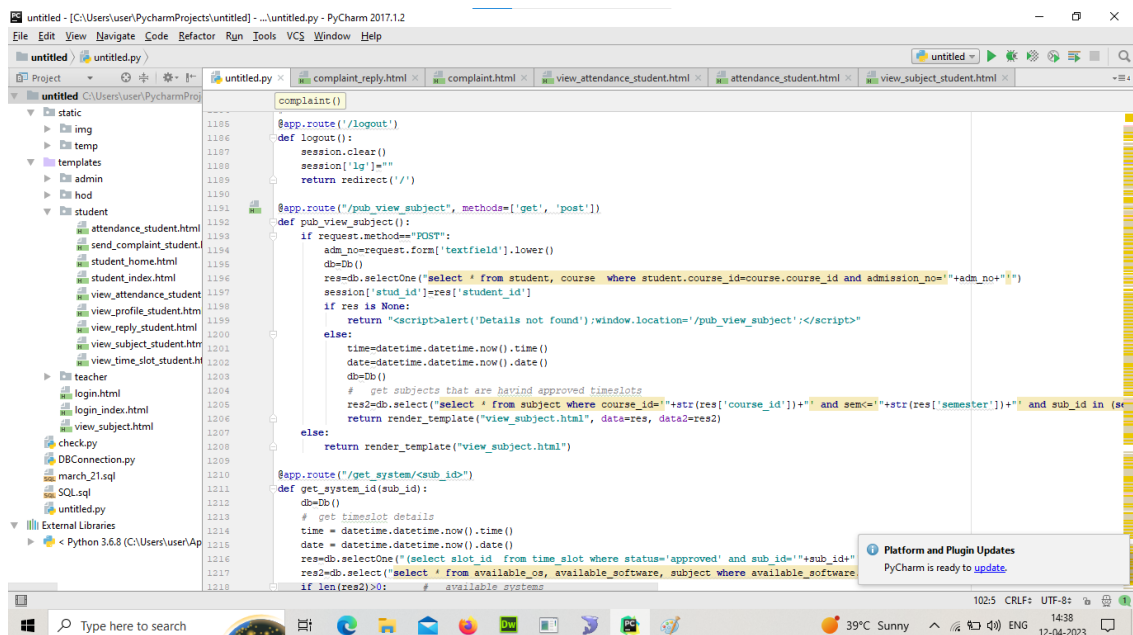


```
untitled - [C:\Users\user\PycharmProjects\untitled] - untitled.py - PyCharm 2017.1.2
File Edit View Navigate Code Refactor Run Tools VCS Window Help

untitled
untitled.py
complaint_reply.html
complaint.html
view_attendance_student.html
attendance_student.html
view_subject_student.html

static
img
temp
templates
admin
hod
student
attendance_student.html
send_complaint_student.html
student_home.html
student_index.html
view_attendance_student.html
view_profile_student.html
view_reply_student.html
view_subject_student.html
view_time_slot_student.html
teacher
login.html
login_index.html
view_subject.html
check.py
DBConnection.py
march_21.sql
SQL.sql
untitled.py
External Libraries
Python 3.6.8 (C:\Users\user\AppData\Local\Programs\Python\Python36-64\python.exe)

complaint()
@app.route('/student_home')
def student_home():
    if session['lg']!='lin':
        return redirect('/')
    return render_template('student/student_index.html')
@app.route('/view_profile_student')
def view_profile_student():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    res = db.selectOne("select * from student,course where student.course_id=course.course_id and student.student_id="+str(session['lid'])+"")
    return render_template('student/view_profile_student.html', data=res)
@app.route('/view_subject_student')
def view_subject_student():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    res = db.select("select * from student,subject where student.course_id=subject.course_id and subject.sem=student.semester and student.student_id="+str(session['lid'])+"")
    return render_template('student/view_subject_student.html', data=res)
@app.route('/view_time_slot_student')
def view_time_slot_student():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    res = db.select("select * from time_slot,subject where time_slot.sub_id=subject.sub_id and time_slot.status='approved' and subject.course_id="+str(session['cid'])+"")
    return render_template('student/view_time_slot_student.html', data=res)
@app.route('/view_attendance_student')
def view_attendance_student():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    res = db.select("select * from time_slot,subject where time_slot.sub_id=subject.sub_id and time_slot.status='approved' and subject.course_id="+str(session['cid'])+"")
    return render_template('student/view_attendance_student.html', data=res)
@app.route('/send_complaint_student')
def send_complaint_student():
    if session['lg']!='lin':
```



```
untitled - [C:\Users\user\PycharmProjects\untitled] - untitled.py - PyCharm 2017.1.2
File Edit View Navigate Code Refactor Run Tools VCS Window Help

untitled
untitled.py
complaint_reply.html
complaint.html
view_attendance_student.html
attendance_student.html
view_subject_student.html

static
img
temp
templates
admin
hod
student
attendance_student.html
send_complaint_student.html
student_home.html
student_index.html
view_attendance_student.html
view_profile_student.html
view_reply_student.html
view_subject_student.html
view_time_slot_student.html
teacher
login.html
login_index.html
view_subject.html
check.py
DBConnection.py
march_21.sql
SQL.sql
untitled.py
External Libraries
Python 3.6.8 (C:\Users\user\AppData\Local\Programs\Python\Python36-64\python.exe)

complaint()
@app.route('/student_home')
def student_home():
    if session['lg']!='lin':
        return redirect('/')
    return render_template('student/student_index.html')
@app.route('/view_profile_student')
def view_profile_student():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    res = db.selectOne("select * from student,course where student.course_id=course.course_id and student.student_id="+str(session['lid'])+"")
    return render_template('student/view_profile_student.html', data=res)
@app.route('/view_subject_student')
def view_subject_student():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    res = db.select("select * from student,subject where student.course_id=subject.course_id and subject.sem=student.semester and student.student_id="+str(session['lid'])+"")
    return render_template('student/view_subject_student.html', data=res)
@app.route('/view_time_slot_student')
def view_time_slot_student():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    res = db.select("select * from time_slot,subject where time_slot.sub_id=subject.sub_id and time_slot.status='approved' and subject.course_id="+str(session['cid'])+"")
    return render_template('student/view_time_slot_student.html', data=res)
@app.route('/view_attendance_student')
def view_attendance_student():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    res = db.select("select * from time_slot,subject where time_slot.sub_id=subject.sub_id and time_slot.status='approved' and subject.course_id="+str(session['cid'])+"")
    return render_template('student/view_attendance_student.html', data=res)
@app.route('/send_complaint_student')
def send_complaint_student():
    if session['lg']!='lin':

logout()
def logout():
    session.clear()
    session['lg']=" "
    return redirect('/')
@app.route("/pub_view_subject", methods=['get', 'post'])
def pub_view_subject():
    if request.method=="POST":
        adm_no=request.form['textfield'].lower()
        db=Db()
        res=db.selectOne("select * from student, course where student.course_id=course.course_id and admission_no="+adm_no+"")
        session['stud_id']=res['student_id']
        if res is None:
            return "<script>alert('Details not found');window.location='/pub_view_subject';</script>"
        else:
            time=datetime.datetime.now().time()
            date=datetime.datetime.now().date()
            db=Db()
            # get subjects that are having approved time slots
            res2=db.select("select * from subject where course_id="+str(res['course_id'])+" and sem="+str(res['semester'])+" and sub_id in (select sub_id from time_slot where status='approved' and course_id="+str(res['course_id'])+"")
            return render_template("view_subject.html", data=res, data2=res2)
        else:
            return render_template("view_subject.html")
@app.route("/get_system/csub_id")
def get_system_id(sub_id):
    db=Db()
    # get timeslot details
    time = datetime.datetime.now().time()
    date = datetime.datetime.now().date()
    res = db.selectOne("select slot_id from time_slot where status='approved' and sub_id="+sub_id+"")
    res2=db.select("select * from available_os, available_software, subject where available_software.sub_id="+sub_id+" and available_software.status='approved'")
    if len(res2)>0:
```

# **CHAPTER VI**

## **SYSTEM TESTING**

## 6.1 TESTING AND EVALUATION

Testing is a process of executing a program with the intent of finding an error. Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design and coding. Testing includes verifications of the basic logic of each program and verification that the entire system works properly. Testing demonstrates that software functions appear to be working according to specification. In addition, data collected as testing is conducted provides a good indication of software quality as a whole. The debugging process is the most unpredictable part of testing process. Testing begins at the module level and works towards the integration of the entire computer-based system. Testing and debugging are different activities, but any testing includes debugging strategy for software testing must accommodate low level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system function, against customer requirements. No testing is complete without verification and validation part. The goals of verification and validation activities are to assess and improve the quality of work products generated during the development and modification of the software. There are two types of verification: life cycle verification and formal verification. Life cycle verification is the process of determining the degree to which the products of the given phase of the development cycle fulfil the specification established during the prior process. Formal verification is the rigorous mathematical demonstration that source code conforms to its specifications. Validation is a process of evaluating software at the end of the software development process to determine conformance with the requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The primary objectives, when we test software are the following:

- Testing is a process of executing with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.
- A successful test is one uncovers undiscovered errors.

Thus, testing plays a very critical role in determining the reliability and efficiency of the software and hence is very important stage in software development. Tests are to be

conducted on the software to evaluate its performance under a number of conditions. Ideally, it should so at the level of each module and also when all of them are integrated to form the completed system. Software testing is done at different levels.

## **6.2 TESTING STRATEGIES**

A strategy for software testing integrates software test case design method into a well-planned series of steps that result in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote a customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project progresses.

**The general characteristics of software testing strategies are:**

- Testing begins at the component level and works “outward” toward the integration of the entire computer system.
- Different testing techniques are appropriate at different points in time.

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

A strategy must provide guidance for the practitioner and set of milestones for the manager. Because the step on the test strategy occurs at a time when deadline pressure begins to rise, progress must be measurable and problem must surface as early as possible.

The software teams approach to testing is defining a plan that describes an overall strategy and a procedure that defines specific testing steps and tests that will be conducted. In the proposed system, if the administrator makes any attempt to login to the application without entering his password, then the system will not allow the user to login to the application.



## **6.3 TESTING TECHNIQUES**

The various testing techniques are given below:

### **6.3.1 WHITE BOX TESTING**

White-box testing is also called as glass-box testing, is a test case design method that goes to the control structure of the procedural design to derive test cases. Using white box testing methods, the software engineer can derive test cases that,

- ✓ Guarantee that all independent paths within a module have been exercised at Least once.
- ✓ Exercise all logical decision on their true and false sides.
- ✓ Execute all loops at their boundaries and within their operational sides.
- ✓ Exercise internal data structure to ensure their validity.

White box testing was successfully conducted on our system. All independent paths within a module have been executed at least once and all logical decisions have been exercised on their true and false sides.

### **6.3.2 BLACK BOX TESTING**

Black-box testing is also called as behavioural testing, focuses on the functional requirement of the software. It is a complimentary approach that is likely uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories.

- ✓ Incorrect or missing functions.
- ✓ Interface errors.
- ✓ Error on data structures or external database access.
- ✓ Behaviour or performance errors.
- ✓ Initialization and termination errors.

Black box testing was successfully conducted on your system. The system was divided into a number of modules and testing was conducted on each module. We have tested the system for incorrect or missing functions, interface and performance errors.

### **6.3.3 UNIT TESTING**

Unit testing comprises the set of tests performed by an individual programmer prior to the integration of the system. Testing removes residual bugs and improves the reliability of the system.

Testing allows the developer to find out the design faults if any, and enable correction if needed. Exhaustive unit testing has to be carried out to ensure the validity of the data. In order to successfully test the entire package, unit testing is carried out. Each module was tested as when it was developed. Thus it proved easier to conduct minute testing operation and correct them then and there.

### **6.3.4 INTEGRATION TESTING**

Bottom-up integration is the traditional strategy used to integrate the component of a software system into a functional whole. Bottom-up integration consists of unit testing, followed by subsystem testing and followed by testing of the entire system. Unit testing has the goal of discovering errors in the individual parts of the system.

Parts are tested in isolation from one another in an artificial environment known as “Test Harness”, which consists of driver programs and data necessary to exercise the modules. Unit testing should be as exhaustive as possible to ensure that each representative case handled by each module has been tested. Unit testing is eased by a system structure that is composed of small loosely coupled modules.

Both control and data interfaces must be tested. Large software system may require several levels of subsystem testing. Lower level subsystems are successively combined to form higher level subsystems. In most software systems, exhaustive testing of subsystem capabilities is not feasible due to the combination complexity of the module interfaces. Therefore, test cases must be carefully chosen to exercise the interfaces in the desired manner.

### **6.3.5 ACCEPTANCE TESTING**

Acceptance testing involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements. It is not unusual for two sets of acceptance test to be run, those developed by the quality group and those developed by the customer.

In addition to the functional and performance tests, stress tests are performed to determine the limitation of the system. For example, a compiler might be tested to determine the effect of the symbol table overflow, or real-time system might be tested to determine the effect of simultaneous arrival of numerous high priority interrupts.

### **6.3.6 OUTPUT TESTING**

Output testing of the proposed system is important since no system could be useful if it does not produce the required output.

The output format on the screen is found to be correct, as the format was designed in the system design phase according to the user needs. For the hard copy also the output comes out as the specified requirements by

The user. Hence output testing doesn't result in any correction on the system.

## **CHAPTER VII**

# **SYSTEM IMPLEMENTATION AND DEPLOYMENT**

Implementation is the process of deploying the new system in the operational environment. Proper implementation is essential to provide a reliable system to meet the organizational requirement. There are four types of implementation methods. They are Direct Changeover, Phased Implementation, Parallel Run and Pilot Approach. The most commonly used implementation methods are Pilot Approach and Parallel Run.

The system which is developed as a web application hence the other functions as normal application, as usual some web development technologies are used in the implementation of the project. The language I selected to program this software is PHP. The reason I selected PHP is that is a simple and powerful language that especially developed to create web application.

Technologies used in the development of the software are:

- ✓ Programming language – Python
- ✓ DBMS – MySQL server
- ✓ Development tool – Py charm
- ✓ Development platform – Windows 10

The front end is HTML, and CSS and back end is MySQL Server and Python. The system developed on Pycharm in Windows 10 operating system.

**CHAPTER VIII**  
**CONCLUSION AND REFERENCE**

## **8.1 CONCLUSION**

The “LARVA” has been developed for all given conditions and it is found working effectively under the all the circumstances that may arise in the real environment. The software has been developed to reduce the paper work.

This system is user friendly and is well efficient to make easy interaction with the users of system. The system is done with an insight into the necessary modifications that may be required in the future. Hence the system can be maintained successfully without much work.

### **8.1.1 FUTURE ENHANCEMENT**

As a future venture, it is suggested to make some changes to provide more services and to provide information at right time in right manner.

The current system is designed to mark attendance and allocate corresponding system to the student. It can only mention the number of the system allocated to the student.

In the future the system can be updated to limiting the access of the computer system by students only to the allocated computer system controlled by the proposed system itself. More features like extending the system to track the activities of the students allocated system. All the functions have been done carefully and successfully in the software, and if any development is necessary in future, it can be done without affecting the design by adding additional modules to the system.

## **8.2 REFERENCE**

- Taming python by programming, Dr. Jeeva Jose
- Fundamentals of database system, Ramez Elmasri, Shamkant B.Navathe
- Youtube
- Google.com

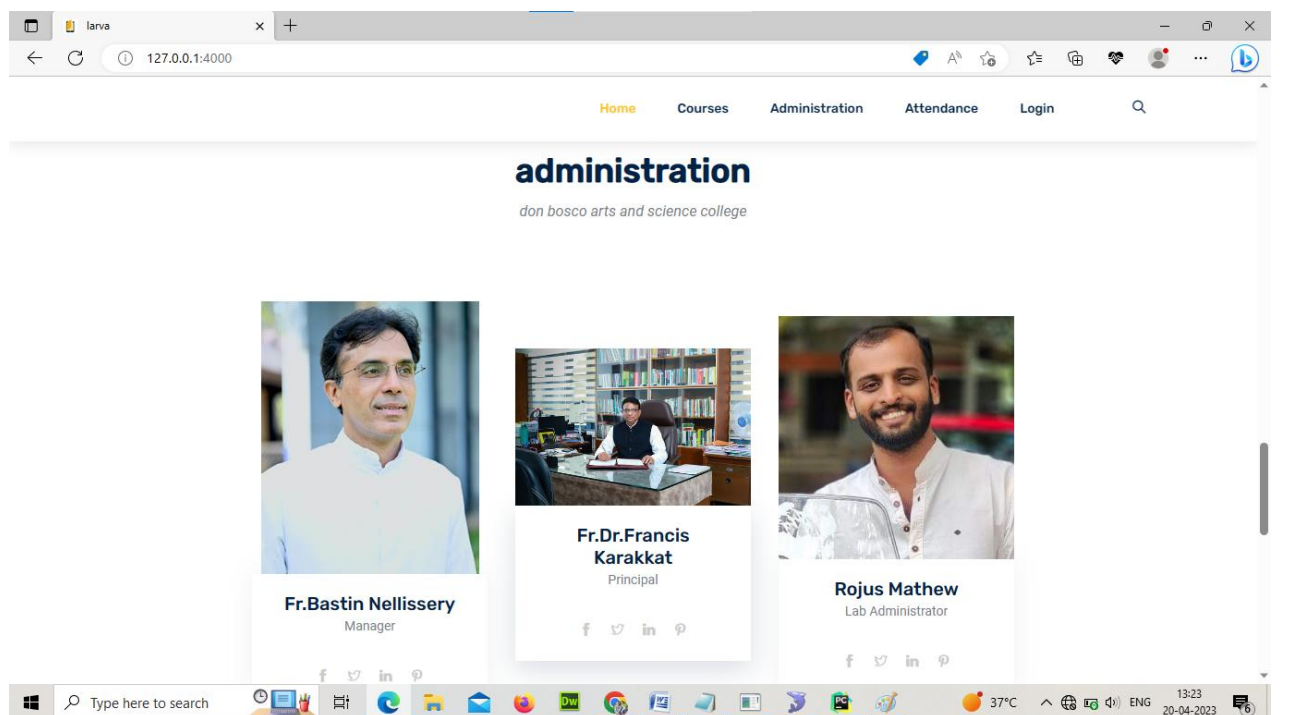
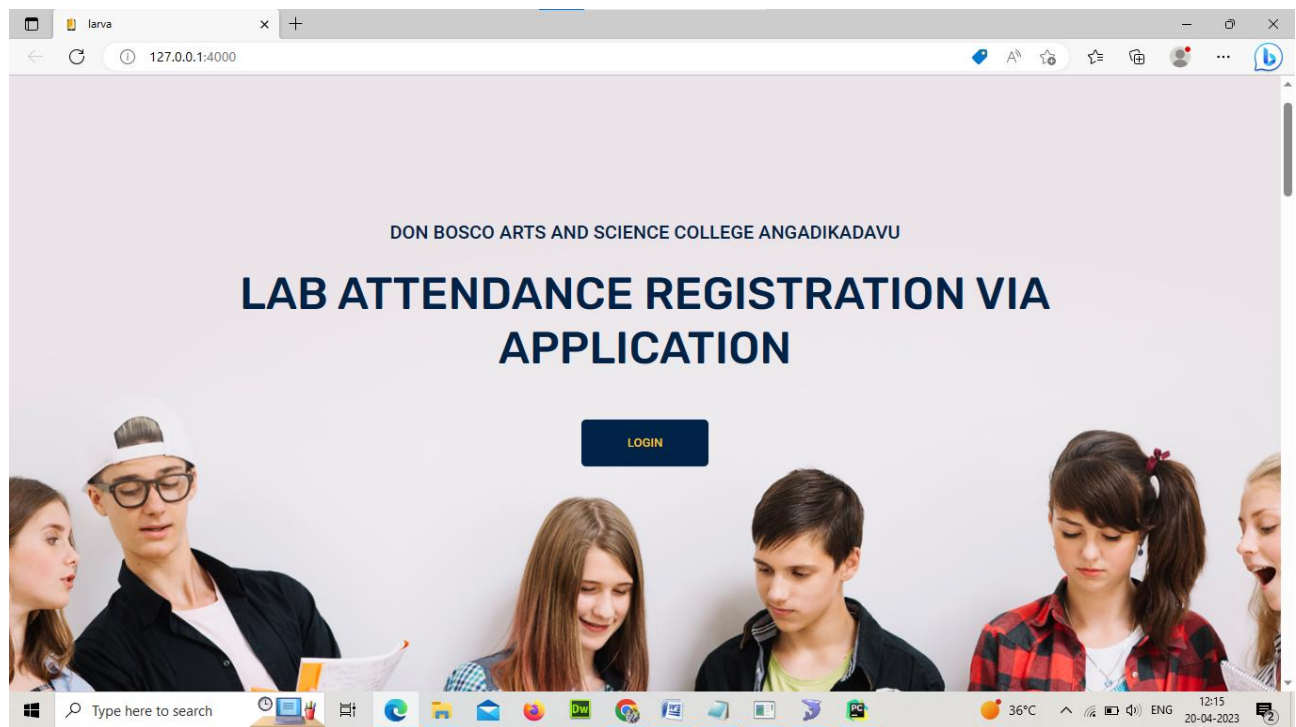
## **CHAPTER IX**

## **APPENDIX**



## 9.1 Screenshots



### 9.1.1 Homepage:



Edustage Education

127.0.0.1:4000/view\_student\_management\_hod#bbb

SUBJECTSTUDENTVIEW PROFILEVIEW COURSEVIEW TEACHERATTENDACE REPORTLOGOUT

#	name	course_id	semester	Register_no	Admission No	date_of_birth	phone	address	email	
1	 Sanjo Jose Augustine	BCA	6	DB20BCAR11	ST3893	05/04/2001	8547188249	place:Kannur post:Poolakutty	sanjojoseaugustin@gmail.com	<div>Edit</div> <div>Delete</div>
2	 Adon Besty	BCA	1	DB20BCAR18	ST4033	01/08/1999	9746653774	place:Iritty post:Palathumkadavu	adonbesty01@gmail.com	<div>Edit</div> <div>Delete</div>

Edustage Education

127.0.0.1:4000/view\_subject\_management\_hod#bbb


SUBJECTSTUDENTVIEW PROFILEVIEW COURSEVIEW TEACHERATTENDACE REPORTLOGOUT

#	COURSE	SEMESTER	SUBJECT	OS	SOFTWARE	
1	BCA	sem1	Informatics	Windows	Python	<div>Edit</div> <div>Delete</div>
2	BCA	sem2	Cpp	Linux	C++	<div>Edit</div> <div>Delete</div>
3	BCA	sem3	Java	Windows	C++	<div>Edit</div> <div>Delete</div>
4	BCA	sem4	sql	Windows	C++	<div>Edit</div> <div>Delete</div>
5	BCA	sem5	EJP	Windows	Python	<div>Edit</div> <div>Delete</div>

larva

127.0.0.1:4000/view\_profile\_teacher#bbb

VIEW PROFILEVIEW ALLOCATED SUBJECTLOGOUT

staff_name	Hebin Layola
department_name	BCA
email	sanjojoseaugustin@gmail.com
phone	8547188249
qualification	MCA
house	uigt
place	Iritty
post	Iritty
Image	

Type here to search

37°C


13:21

20-04-2023

larva

127.0.0.1:4000/view\_allocate\_subject\_teacher#bbb

VIEW PROFILEVIEW ALLOCATED SUBJECTLOGOUT



#	sem	course	subject		
1	sem1		Informatics	<div>Request_for_time_slot</div> <div>View_time_slot</div>	<div>students</div> <div>generate report</div>

Type here to search

37°C

13:22

20-04-2023

**A PROJECT REPORT ON**  
**MENU ASSIST**

Submitted in partial fulfilment of the requirement for award of the degree.

Of  
**Bachelor of Computer Application**  
Of  
**KANNUR UNIVERSITY**

By  
**ADHISH P**  
**REG.NO: DB20BCAR02**  
**ALBIN SUNNY**  
**REG.NO: DB20BCAR04**  
**DAYAN JOSEPH**  
**REG.NO: DB20BCAR27**



**DON BOSCO ARTS & SCIENCE COLLEGE**  
**ANGADIKADAVU, KANNUR, 670706**  
**2023**

**A PROJECT REPORT ON**  
**MENU ASSIST**

Submitted in partial fulfilment of the requirement for award of the degree.

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ADHISH P**

**REG.NO: DB20BCAR02**

**ALBIN SUNNY**

**REG.NO: DB20BCAR04**

**DAYAN JOSEPH**

**REG.NO: DB20BCAR27**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2022**

**DON BOSCO ARTS & SCIENCE COLLEGE**

# ANGADIKADAVU IRITTY, KANNUR



## CERTIFICATE

Certified that this report titled **MENU ASSIST** is a Bonafede record of the project work done by **Adhish P (Reg.No: DB20BCAR02)**, **Albin sunny(Reg.No:DB20BCAR04)** and **Dayan joseph (Reg.No:DB20BCAR27)** under the supervision and guidance ,towards partial fulfilment of the requirement for award of the degree of bachelor of computer application (BCA) of the Kannur university.

Project Guide

Head of the Department

Angadikadavu

External Examiner

Date:

- 1.
- 2.

# Declaration

We ADHISH P , ALBIN SUNNY and DAYAN JOSEPH sixth semester BCA student of Don Bosco Arts & Science College, Angadikadavu, under Kannur University do hereby declare that the project entitled **MENU ASSIST** is the original work carried out by me in the sixth semester under the supervision of Mr. HEBIN LAYOLA, Lecturer of the Department of BCA, Don Bosco Arts & Science College, Angadikadavu, in partial fulfilment of the requirement for the award of the degree Bachelor of Computer Application, Kannur University.

Angadikadavu

Date

ADHISH P

ALBIN SUNNY

DAYAN JOSEPH

# ACKNOWLEDGEMENT

First of all we thank the lord almighty for his immense grace and blessings showered on me at every stages of this work. I am greatly indebted to our Principal Fr. Dr. Francis Karackat SDB, Don Bosco Arts & Science College, Angadikadavu for providing the opportunity to take up this project as part of my curriculum.

We deeply indebted to my project guide Mr. Hebin Layola, lectures of department of BCA, for her assistance and valuable suggestions as guide. She made this project a reality.

We express our sincere thanks to Mrs. Sindu PM, Ms. Sruthi Nimesh, Mrs. Vineetha Mathew and Mrs. Fincy Cyriac, lecturers of department of BCA, for their valuable suggestions during the course of this project. Their critical suggestions helped me to improve the project work.

Acknowledging the efforts of everyone, their chivalrous help in the course of the project preparation and their willingness to collaborate with the work, their magnanimity through lucid technical details lead to the successful completion of my project.

We would like to express my sincere thanks to all my friends, colleagues, parents and all those who have directly or indirectly assisted during this work.



# CONTENTS

<b>chapters</b>	<b>contents</b>	<b>Page No</b>
1	Introduction	1
2	System Analysis	7
2.1	Existing System	8
2.1.1	Disadvantage Of Existing System	8
2.2	Proposed System	9
2.2.1	Advantage Of Proposed System	9
2.3	Feasibility Study	9
2.3.1	Economic Feasibility	10
2.3.2	Technical Feasibility	10
2.3.3	Behavioural Feasibility	11
2.4	System Specification	11
2.4.1	Software Specification	11
2.4.2	Hardware Specification	12
2.5	Identification Of Actors	12
2.6	Identification Of Use Cases	13
2.6.1	Use Cases for The Actor Admin	13
2.6.2	Use Cases for The Actor Kitchen	14
2.6.3	Use Cases for The Actor Service Station	15
2.6.4	Use Cases for The Actor Cashier	16
2.6.5	Use Cases for The Actor Customer	16

2.6.6	Use Case Diagram	18
3	SYSTEM DESIGN	21
3.1	Introduction	22
3.2	Database Design	22
3.3	Table Design	23
3.4	Menu Assist	24
3.5	Data Flow Diagram	28
3.6	ER Diagram	37
4	CODING	39
4.1	Input Interface	39
4.2	Output Interface	39
4.3	Software Description	39
4.3.1	HTML	39
4.3.2	CSS	40
4.3.3	JavaScript	42
4.3.4	MySQL	42
4.3.5	Python	45
4.3.6	Flask	46
5	CODING PAGES	47
5.1	Admin Page	48
6	System Testing	60

6.1	Testing And Evaluation	61
6.2	Testing Strategies	62
6.3	Testing Techniques	63
6.3.1	White Box Testing	63
6.3.2	Black Box Testing	64
6.3.3	Unit Testing	64
6.3.4	Integration Testing	64
6.3.5	Acceptance Testing	65
6.3.6	Output Testing	65
7	SYSTEM IMPLEMENTATION AND DEPLOYMENT	66
8	CONCLUSION	67
9	REFERANCE	68
10	APPENDIX	69
10.1	Website	69
10.2	Android	82



# **1.INTRODUCTION**

## **1.1 Project Overview**

The “Menu assist” is a new, revolutionary electronic menu system for restaurants inside a mall or a shopping complex. Which allows users to place orders with ease. Instead of traditional paper menus, provide your customers with digital restaurant menus! From an iPad or other tablet, customers can browse all your menu items (plus images) in a clean, easy-to-use format. Once they decide what they want, they don’t even have to wait for someone to come take their order– customers can place orders straight from the tablet. Once placed, all orders go directly to the kitchen. How’s that for quick service?

The Menu server PC includes a touch-screen monitor, which allows waiters to quickly submit all incoming orders. This saves them time as well as minimising potential order errors. Food orders can be set up to print straight to the kitchen. All printing settings are fully customisable to your particular restaurant.

All order history is recorded according to the time the order is places. We offer options to allow multiple payments as well as bill splitting. We also record history of all table information so that any particular day’s summary can be viewed and printed at any time, making tax time a breeze.

### **NEED FOR THE SYSTEM**

By implementing an electronic menu system like Menu assist, restaurants in malls or shopping complexes can address these challenges and provide a more streamlined and convenient experience for their customers. The system can allow customers to easily browse menus, place orders, and make payments from their mobile devices, reducing wait times and improving efficiency. Additionally, the system can provide customers with more information about menu items, dietary restrictions, and special offers, leading to a more personalized and satisfying experience.

The Menu assist system can also benefit restaurant owners and employees by providing them with a customizable solution that can be tailored to their specific needs and branding. The system can help them to manage staff, menus, tables, and venues more efficiently, while also providing them with valuable data and reports to help them optimize their operations and improve their business performance.

## **OBJECTIVES AND SCOPE**

The electronic menu system is highly customisable to suit your restaurant's unique needs and desires. Our tablet menu system for restaurants can be used in a variety of different ways, by both customers and employees.

The Menu Assist also provides the feature that the customer can view details of other shops or activities that are offered within the mall or shopping complex. So, the customer can effectively use the waiting time and thereby they can reduce the time spent for searching a particular shop or offerings inside the mall.

The main objectives behind the development of this project are:

- 1) Improve the overall customer experience by making it easier and more convenient for customers to browse menus, place orders, and receive their food.
- 2) Increase efficiency and productivity by streamlining the ordering process and reducing wait times for customers.
- 3) Provide restaurant owners and employees with a customizable system that can be tailored to meet their specific needs and branding.

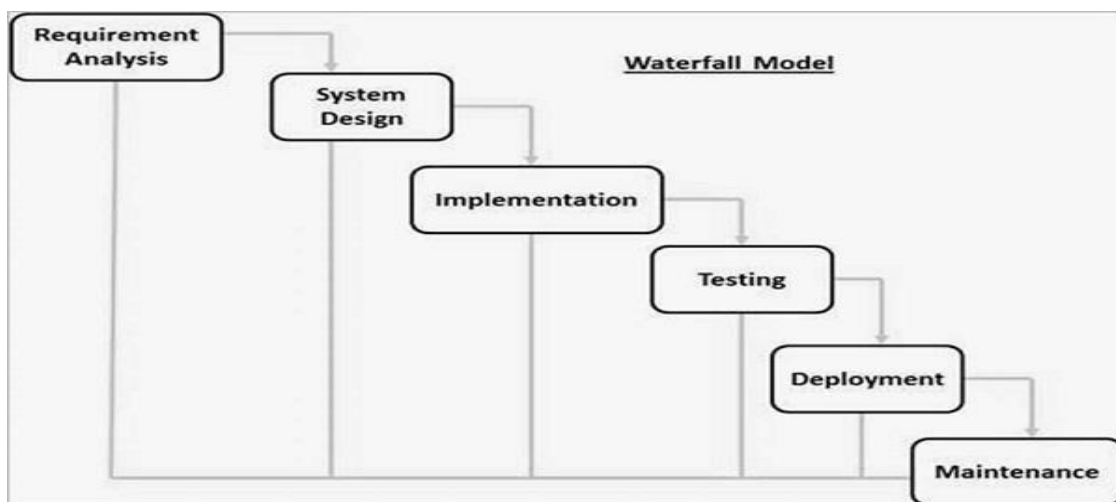
The scope of the electronic menu system is focused on enhancing the overall customer experience, increasing efficiency, and providing a customizable solution for restaurant owners and employees in malls or shopping complexes.

## **MODEL:**

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially

### **Waterfall Model - Design:**

In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. Following is the pictorial representation of Iterative and Incremental model:



The sequential phases in Waterfall model are –

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.



- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

### **Waterfall Model - Application:**

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are –

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

**The advantages of the waterfall model SDLC Model are as follows:**

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Some of the major advantages of the Waterfall Model are as follows –

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

**The disadvantages of the waterfall model SDLC Model are as follows:**

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows –

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.

- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

## **2. SYSTEM ANALYSIS**

## **Introduction**

System analysis is the process of collecting and interpreting facts, understanding problems and using the information to suggest improvement on the system. This will help to understand the existing system and determine how computers make its operation more effective. The aim of this analysis is to collect detailed information on the system and the feasibility study of the proposed system.

## **2.1 EXISTING SYSTEM**

Restaurant services such as making reservations, processing orders, and delivering orders generally require waiters to input customer information and then transmit the orders to kitchen for the preparation. When the customer pays the bill, the amount due is calculated by the cashier. Although this procedure is simple, it may significantly increase the workload of waiters and even cause errors in food ordering or in prioritizing customers, especially when the number of customers suddenly increases during busy hours, which can seriously degrade the overall service quality.

### **2.1.1 The Disadvantages of Existing System**

The existing system has the following disadvantages,

- ° The manual process of taking orders, transmitting them to the kitchen, and calculating bills can be time-consuming and increase the workload for waitstaff, especially during peak hours.
- ° increase the potential for errors, which can lead to delays in service and dissatisfied customers.

## **2.2 PROPOSED SYSTEM**

The user can then browse the menu however they want to, sorting the items on various dimensions like price, popularity, ratings, etc. The user can also click through to view more information about any item like nutritional information, ingredients, trivia etc. While browsing through the menu, the customer may add items to his/her order, the user can edit and place the order. The staff will automatically and almost instantly be notified about the new order so that they can act on it. The user may even track the status of their order so that the customer may know when to expect their food and drinks to their table.

### **2.2.1 Advantages of Proposed System**

- Increase the efficiency of the restaurant.
- Ensure that customers receive their orders accurately and promptly.
- Reduce wait times for customer.
- Enhance the overall dining experience for customer.
- Improve the overall customer experience and increase customer satisfaction.

## **2.3 Feasibility Study**

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus, when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

The document provides the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioural feasibilities.

The proposed system is theoretically investigated to check the feasibility and found that they are more reliable and efficient in the cases given below. There are three aspects in the feasibility study portion of the preliminary investigation.

✓ Economic feasibility

✓ Technical feasibility

✓ Behavioural feasibility

The proposed system must be evaluated from a technical point of view first,

and if technical feasible their impact on the organization must be assessed. If compatible, the operational system can be devised. Then they must be tested for economic feasibility.

### **2.3.1 Economic Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors which affect the development of a new system is the cost it would require. Since the system developed as part of project work, there is no manual cost to spend for the proposed system. Also, all the resources are already available, it gives an indication of the system is economically possible for development.

### **2.3.2 Technical Feasibility**

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, procedures and staff. Having identified an outline system, the investigation must go on suggest the type of equipment, required method developing the system, of running the system once it has been designed. The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Though the technology become obsolete after some period of time, due to the fact that newer version of some software supports older versions, the system still be used. So, there are only minimal constraints involved with this project. The system has been developed using C# and .NET, along with the database software SQL server, thus we could conclude that the project is technically feasible for development.

### **2.3.3 Behavioural Feasibility**

People are inherently resistant to change and computers have been known to facilitate change. The System is designed in user friendly manner and we need to provide any special training for the persons using this software. The operating system used is Windows 10, which is also user friendly. Since the application is web biased and can easily accessed in a web browser, which is quite familiar to the intended users, it does not have any operational barriers. So, no need to provide any special training for using this application software and hence it is behaviourally feasible.

## **2.4 System Specifications**

System Specification deals with the technical aspects the project has to meet in minimum to work successfully. This also includes the different aspects the software requirement is determined from. The technical details typically include:

- Software Specification
- Hardware Specification

### **2.4.1 Software Specifications**

The software required for the application depends on the following factors:

- ✓ The flexibility of the software
- ✓ Software contracts
- ✓ Limitation of the software

## **Software Requirement**

This specifies the minimum software requirements for implementing the system. This includes:

- Front End: - HTML,java.
- Back End: - MySQL
- Client-side scripting: java script
- Server-side scripting: Python
- Platform: Flask, JetBrains PyCharm, Android Studio
- Operating System: Microsoft windows 7 or above

### **2.4.2 Hardware Specifications**

The software required for the application depends on the following factors:

- ✓ Determining size and capacity requirements.
- ✓ Computer evaluation and measurement.
- ✓ Financial factors.
- ✓ Maintenance and support.

### **Hardware Requirement**

- Microprocessor: Any 64-bit processor.
- Processor: Intel core i3 or above
- Clock speed: - 2.13GHz
- Ram: 4 GB or above
- Hard disk: 40 GB above free space
  
- Keyboard: -standard keyboard
- Mouse: Standard mouse
- Connectivity: - LAN & Wi-Fi
- Camera: Standard Camera

## **2.5 Identification of Actors**

A use cases represents the functionality of an actor. It is defined as a set of actions performed by a system, which yields an observable result. An ellipse containing its name inside the ellipse or below it represents. it. It is placed inside the system boundary and connected to an actor with an association. This shoes how the use cases and the actor interact.



We can identify the actors through a list of questions. The answers to these questions bring out the actors of the system is.

- Admin
- Kitchen
- Service Station
- Cashier
- Customer

Here we need to specify the use cases of each actor.

## **2.6 Identification of use cases**

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yield an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

To find out the use cases, ask the following questions to each of the actors.

- ✓ Which functions does the actor require from the system? What does the actor need to do?
- ✓ Does the actor need to read, create, destroy, modify or store some kind of information in the system?
- ✓ Does the actor have to calculate something? And want to provide information for others?
- ✓ Could the actor's daily work be simplified or made more efficient by adding new functions to the system (typically functions which are currently not automated in the system)?

### **2.6.1 Use cases for the actor Admin**

#### **Login:**

The first step involved is login. The admin can login to the website using username and password.

**Staff Management:**

Admin can add or remove staff.

**Menu Management:**

Admin can add or remove food menu.

**Today's Special Management:**

Admin can add or remove today's special items.

**Table Management:**

Admin can manage table.

**View User:**

Admin can view user.

**View Complaints and Send Reply:**

Admin can view complaints and send its reply.

**View complaint and replay:**

Admin can view complaints and replay.

**View Report:**

Admin can view report.

**Venue Category Management:**

Admin can add or remove venue category.

**Venue Category:**

Admin can add or remove venue.

**2.6.2 Use cases for the actor Kitchen****Login:**

The Kitchen can login to the website using username and password.

**View New Order and Update Status:**

Kitchen can view new order and update its status.

**Add Time Delay Details:**

Kitchen can add time delay details of orders.

**View Reviews from Customer:**

Kitchen can view reviews from customer.

**2.6.3 Use cases for the actor Service Station****Login:**

Service Station can login to the website using username and password.

**View Completed Order:**

Service Station can view the completed order.

**Allocate to Serving Staff:**

Service Station can allocate serving staff.

**Update Order Status:**

Service Station can update order status.

**2.6.4 Use cases for the actor Cashier****Login:**

Cashier can login to the website using username and password.

**View Bill:**

Cashier can view bill.

**View Payment Report:**

Cashier can view payment report.

### **2.6.1 Use cases for the actor Customer**

#### **Register:**

#### **Login:**

The first step involved is login. The customer can login to the app using username and password.

#### **View Menu:**

Customer can view food menu.

#### **View Today's Special Item:**

Customer can view today's special items.

#### **View Reviews:**

Customer can view reviews of items.

#### **Add to cart:**

Customer can add items into cart.

#### **Check Out Items:**

Customer can check out items.

#### **View Order Status:**

Customer can view order status.

#### **View Bill:**

Customer can view bill.

#### **Payment:**

Customer can pay their order.

**Send Review:**

Customer can send review.

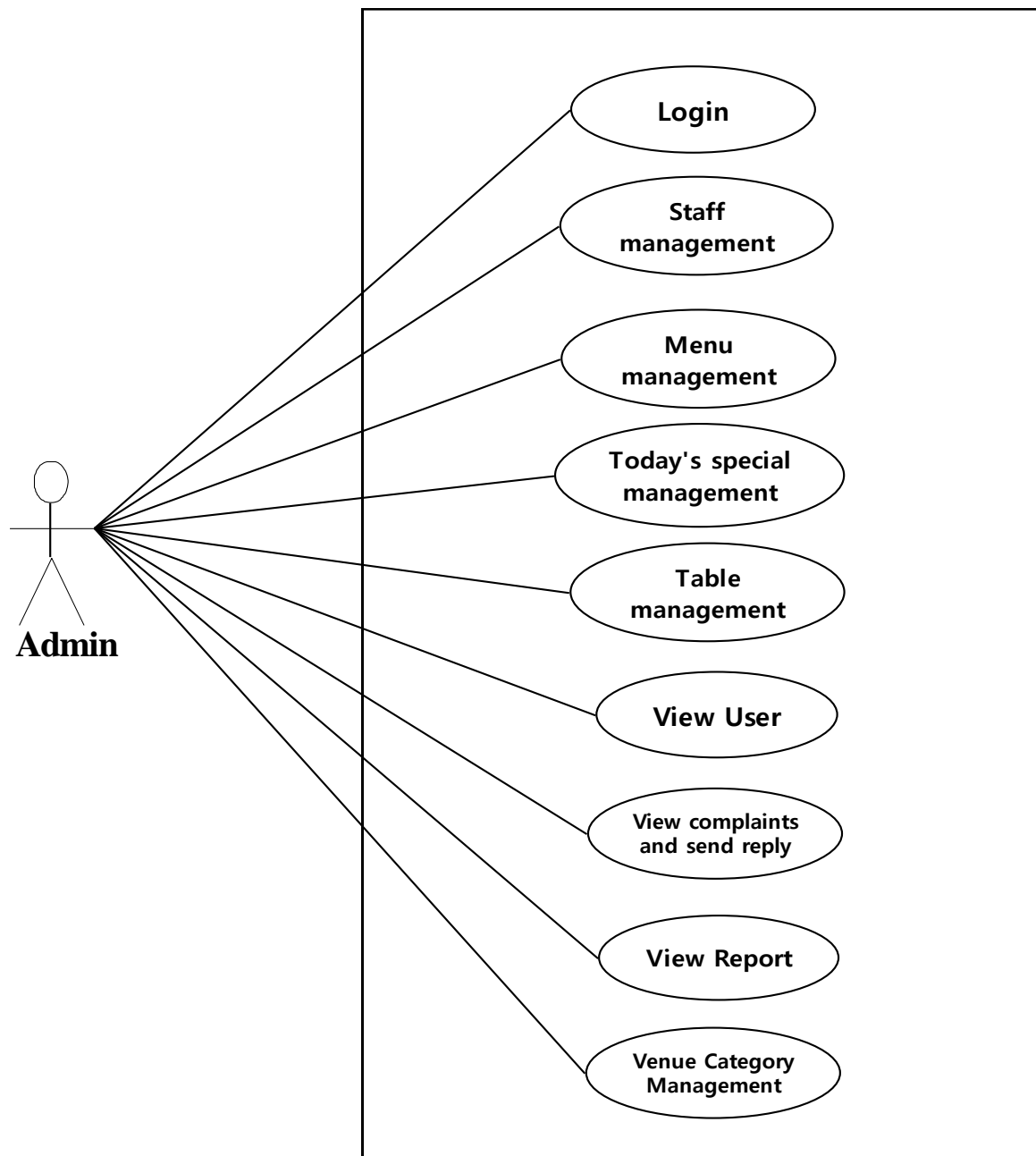
**View Venue Category:**

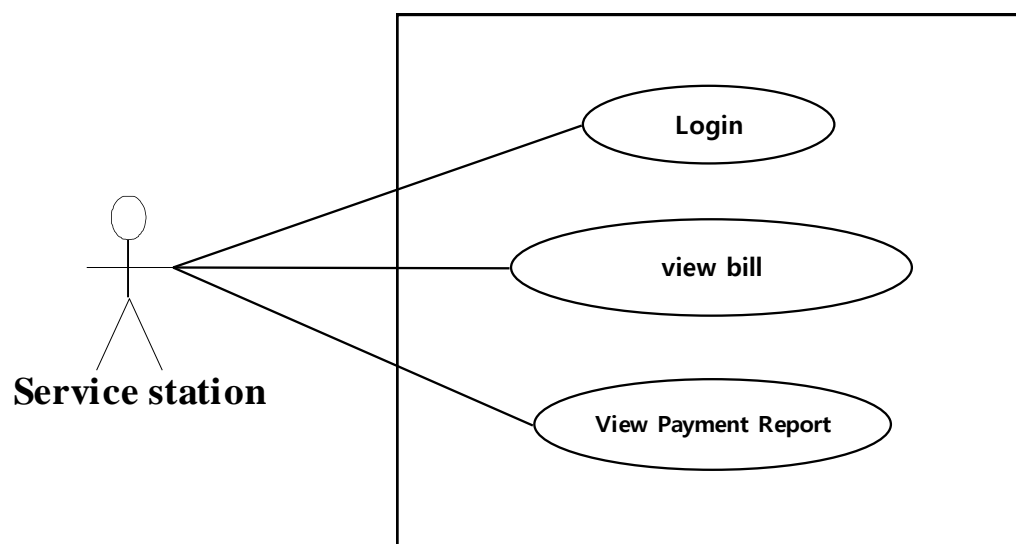
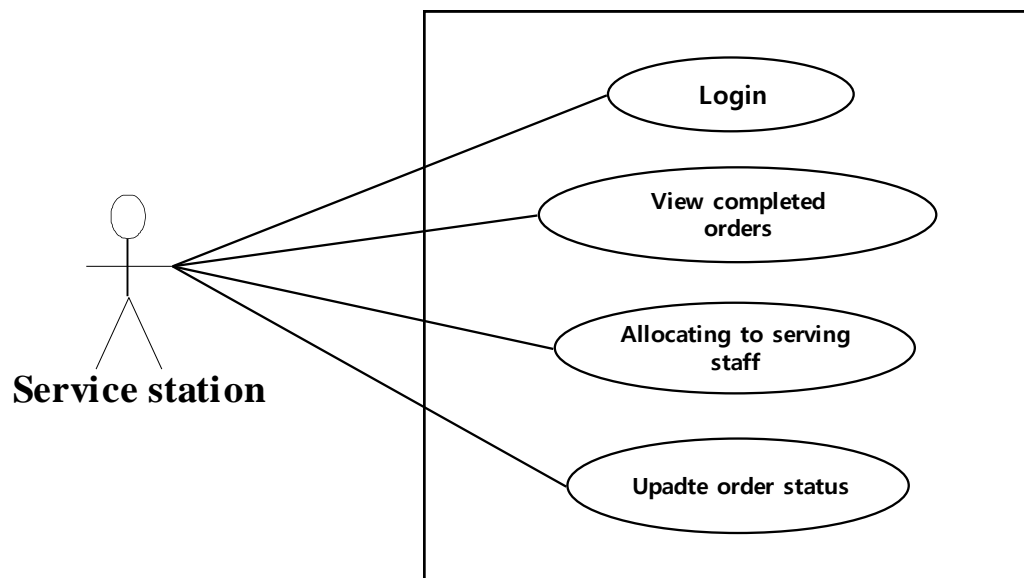
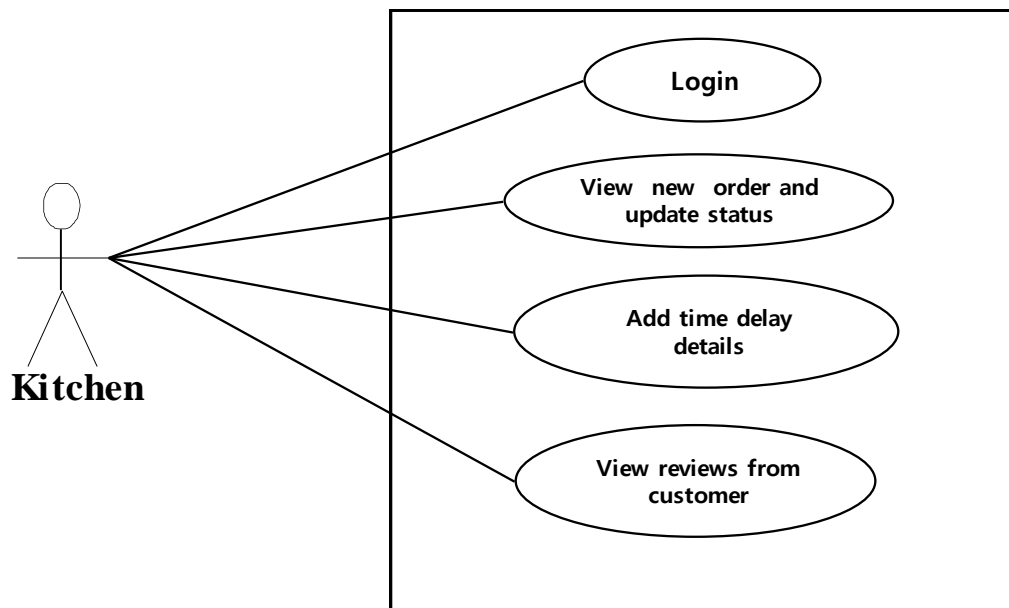
Customer can view venue category.

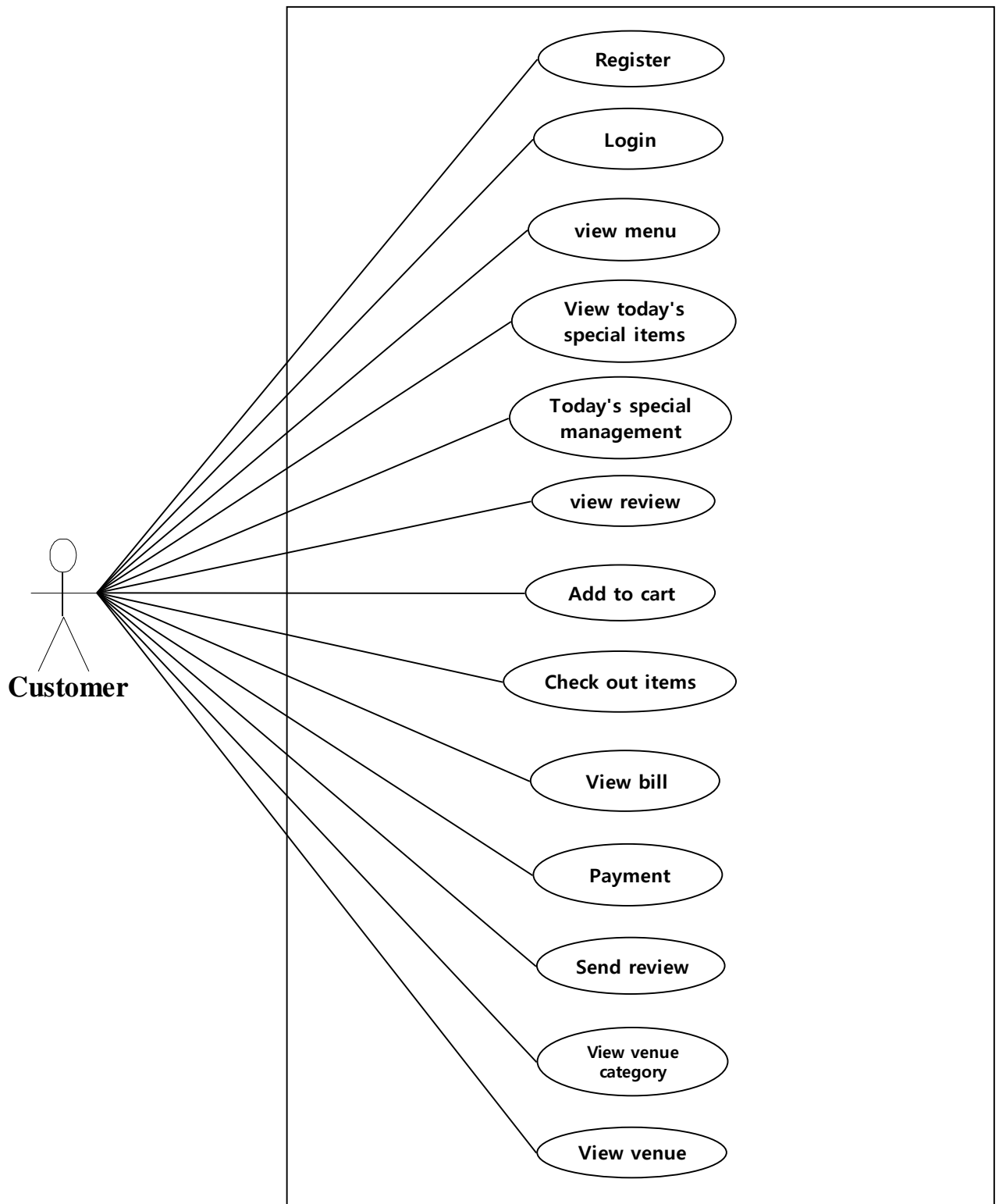
**Venue Category:**

Customer can view venue.

## 2.6.6 USE CASE DIAGRAM









### **3. SYSTEM DESIGN**

### **3.1 INTRODUCTION:**

System design provides an understanding of the procedural details, necessary implementing the system recommended in the feasibility study. Basically, it is all about the creation of a new system. This is a critical phase since it decides the quality of the system and has a major impact on the testing and implementation phases.

**System design consists of three major steps.**

- Drawing of the expanded system data flow charts to identify all the processing functions required.
- The allocation of the equipment and the software to be used.
- The identification of the test requirements for the system.

### **CHARACTERS OF DESIGN**

- A design should exhibit a hierarchical organization that makes intelligent use of control among components of the software.
- A design should be modular that is, the software should be logical.
- A design should contain distinct and separable representation of data and procedure.
- A design should lead to interface that reduce the complexity of the connections between modules and with the external environment.

### **3.2 Database Design**

A Database is a collection of inter related data stored with minimum redundancy to serve many users quickly and efficiently. In database design data independence, accuracy, privacy and security are given higher priority. Database design is an integrated approach to the file design. This activity deals with the design of the physical data base. All entities and attributes have been identified while creating the database. The database design deals with the grouping of data into number of tables so as to.

- ✓ Reduplication of data.
- ✓ Minimize storage space.

- ✓ Retrieve the data efficiently.

Following are some guidelines for the database design:

- Design a relational schema so that it is easy to explain its meaning. Do not combine attributes from multiple entity and relationship type into a single relation.
- Design the database schema so that no insertion, deletion or modification anomalies are present in the relation.
- As far as possible, avoid placing attributes in the base relation whose values may frequently be null.
- Design relation schema so that they can be joined with equality conditions on attributes that are either primary keys or foreign keys in a way that no spurious tuples are generated.

### 3.3 Table Design

DB design is required to manage large bodies of information. The management of data involves both the definition of the structure of storage of information and provisions of mechanism for the manipulation of information. For developing efficient database certain conditions have to be fulfilled such as:

- Control Redundancy
- Ease of Use
- Data Independence
- Accuracy and Integrity

There are five major steps in design process:

- Identify the table and relationship.
- Identify the data that is needed for each table and relationship.
- Resolve the relationship.
- Verify the design.
- Implement the design

The Database Consist of the following tables given below.

### 3.4 MENU ASSIST

#### *1.Login table*

Colum name	Data type	Constraints	Description
login_id	int	primary key	unique identifier
username	varchar (50)	not null	name of user
password	varchar (50)	not null	secret key
type	varchar (50)	not null	To specify the role

#### *2. Bank table*

Colum name	Data type	Constraints	Description
bank_id	int	primary key	unique identifier
bank_name	varchar (50)	not null	Name of bank
acc_no	varchar (50)	not null	
ifsc	varchar (50)	not null	
amount	Varchar (100)	Not null	

#### *3. Category table*

Colum name	Data type	Constraints	Description
c_id	int	primary key	unique identifier
category	varchar (50)	Not null	

#### *4.Complaint table*

Colum name	Data type	Constraints	Description
complaint_id	int	primary key	unique identifier
user_id	int	foreign key	User id
date	date	not null	Date of complaint
complaint	varchar (200)	not null	

replay_date	date	not null	Date of reply
replay	varchar (200)	not null	

### ***5. Help table***

Column name	Data type	Constraints	Description
help_id	int	primary key	unique identifier
login_id	Int	foreign key	Login id
help	varchar (200)	not null	
table_id	Int	foreign key	
date	Date	not null	Date of
status	Varchar (100)	Not null	

### ***6. Item table***

Column name	Data type	Constraints	Description
item_id	int	primary key	unique identifier
Item_cid	int	foreign key	Item category id
item_name	varchar (100)	not null	Name of item
item_details	varchar (200)	not null	Details of item
item_photo	varchar (100)	not null	photo of item
item_price	int	not null	Price of item

### ***7.Order\_sub table***

Column name	Data type	Constraints	Description
order_id	int	primary key	unique identifier
omaster_id	int	foreign key	Ordered master id
item_id	int	foreign key	Item id
qty	int	Not null	Quantity of item
ostatus	varchar (100)	Not null	
item_type	varchar (100)	Not null	Type of item

### ***8. Order\_master table***

Colum name	Data type	Constraints	Description
master_id	Int	primary key	unique identifier
user_id	Int	foreign key	User id
table_id	int	foreign key	Table id
date	date	Not null	Date
status	varchar (100)	Not null	Status of order

### ***9. Review table***

Colum name	Data type	Constraints	Description
review_id	int	Primary key	unique identifier
User_id	int	foreign key	User id
Item_id	int	foreign key	Item id
Food type	varchar (100)	Not null	Type of food
review	varchar (200)	Not null	
date	date	Not null	date

### ***10.Staff table***

Colum name	Data type	Constraints	Description
staff_id	int	Primary key	login identifier
staff_name	Varchar (200)	foreign key	Name of staff
staff_hn	Varchar (200)	Not null	House name of staff
staff_place	Varchar (200)	Not null	Place of staff
staff_pin	int		Pincode of staff
staff_post	Varchar (200)		Post office of staff
staff_district	Varchar (200)		District of staff
staff_qualification	Varchar (200)		Qualification of staff
staff_photo	Varchar (200)		Photo of staff
staff_email	Varchar (200)		Email of staff
staff_phone	int		

***11. Order\_master table***

Colum name	Data type	Constraints	Description
table_id	Int	primary key	unique identifier
table_no	Int		Table number

***12. today's\_special table***

Colum name	Data type	Constraints	Description
ts_special_id	Int	primary key	unique identifier
ts_category	Varchar (200)	Not null	Today's special category
ts_name	Varchar (200)	Not null	
ts_details	Varchar (200)	Not null	
ts_photo	Varchar (200)	Not null	
ts_price	Varchar (200)	Not null	
ts_date	Date	Not null	Date

***13. User table***

Colum name	Data type	Constraints	Description
user_id	Int	foreign key	User id
user_name	Varchar (200)		Username
email	Varchar (200)		email
photo	Varchar (200)	Not null	photo

***14. Venue\_category\_mngt table***

Colum name	Data type	Constraints	Description
category_name	Varchar (200)		Category name
category_id	Int	foreign key	Category id

### ***15. Venue\_mngmnt table***

Colum name	Data type	Constraints	Description
venue_id	Int	primary key	unique identifier
category_id	Int	foreign key	Category id
venue_name	Varchar (200)		
phon_no	Int		Date
floor	Int		

### **3.5. Data Flow Diagram**

A graphical representation is used to describe and analyses the movement of data through a system manual or automated including the processes, Storing of data and delays in the system. Data flow diagrams are the central tool and the basis from which other components are developed.

The transformation of data, from input to output through process may be described logically and independently of the physical components associated with the system.

They are termed logical dataflow diagrams, showing the actual implementations and the movement of data between people, departments and

workstations. DFD is one of the most important modelling tools used in system design. DFD shows the flow of data through different process in the system.

#### **PURPOSE:**

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have reasonably completed model of the behavior of the system. It is important to avoid premature designs, wherein develop designs before



analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

Throughout my project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In my opinion, "efficient design of the data flow and context flow diagram helps to design the system successfully without much major flaws within the scheduled time". This is the most complicated part in a project. In the designing process, my project took more than the activities in the software lifecycle. If we design a system efficiently with all the future enhancements the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as a way of expressing system requirements in graphical form. A data flow diagram also known as "bubble chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It functionality decomposes the requirement specification down to the lowest level. Data Flow Diagram depicts the

information flow, the transformation flow and the transformations that are applied as data move from input to output. Thus DFD describes what data flows rather than how they are processed.

Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is one of the most important tools used during system analysis. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

Data Flow Diagrams are made up of a number of symbols, which represents system components. Data flow modelling method uses four kinds of symbols, which are used to represent four kinds of system components.

These are

- Process
- Data stores
- Data flows
- External entity

**Process:**

Process shows the work of the system. Each process has one or more data inputs and produce one or more data outputs. Processes are represented by rounded rectangles in Data Flow Diagram. Each process has a unique name and number. This name and number appears inside the rectangle that represents the process in a Data Flow Diagram.

**Data Stores:**

A data stores is a repository of data. Processes can enter data, into a store or retrieve the data from the data store. Each data has a unique name.

**Data Flows:**

Data flows show the passage of data in the system and are represented by lines joining system components. An arrow indicates the direction of flow and the line is labelled by name of the dataflow.

**External Entity:**

External entities are outside the system but they either supply input data into the system or use other systems output. They are entities on which the designer has control. They may be an organizations customer or other bodies with which the system interacts. External entities that supply data into the system are sometimes called source. External entities that use the system data are sometimes called sinks. These are represented by rectangles in the

Data Flow Diagram.

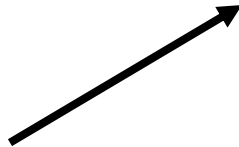
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Basic data flow diagram symbols are.....

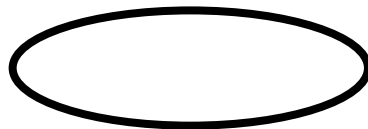
- A Square defines a source (originator) or destination of a system data:



- An Arrow identifies data flow. It is a pipeline through which information flows:



- A Circle represents a process that transforms incoming data flow(s) into outgoing data flow(s):



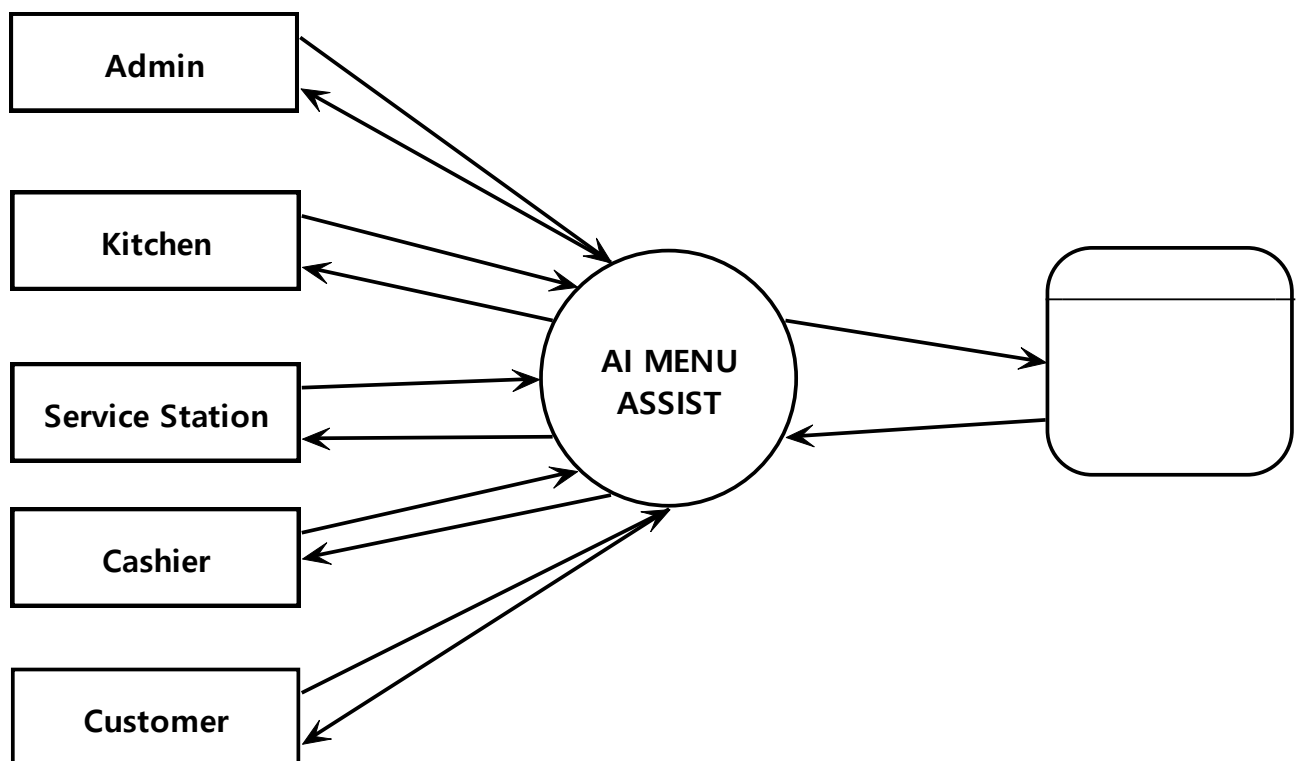
- An Open Rectangle is a data store:



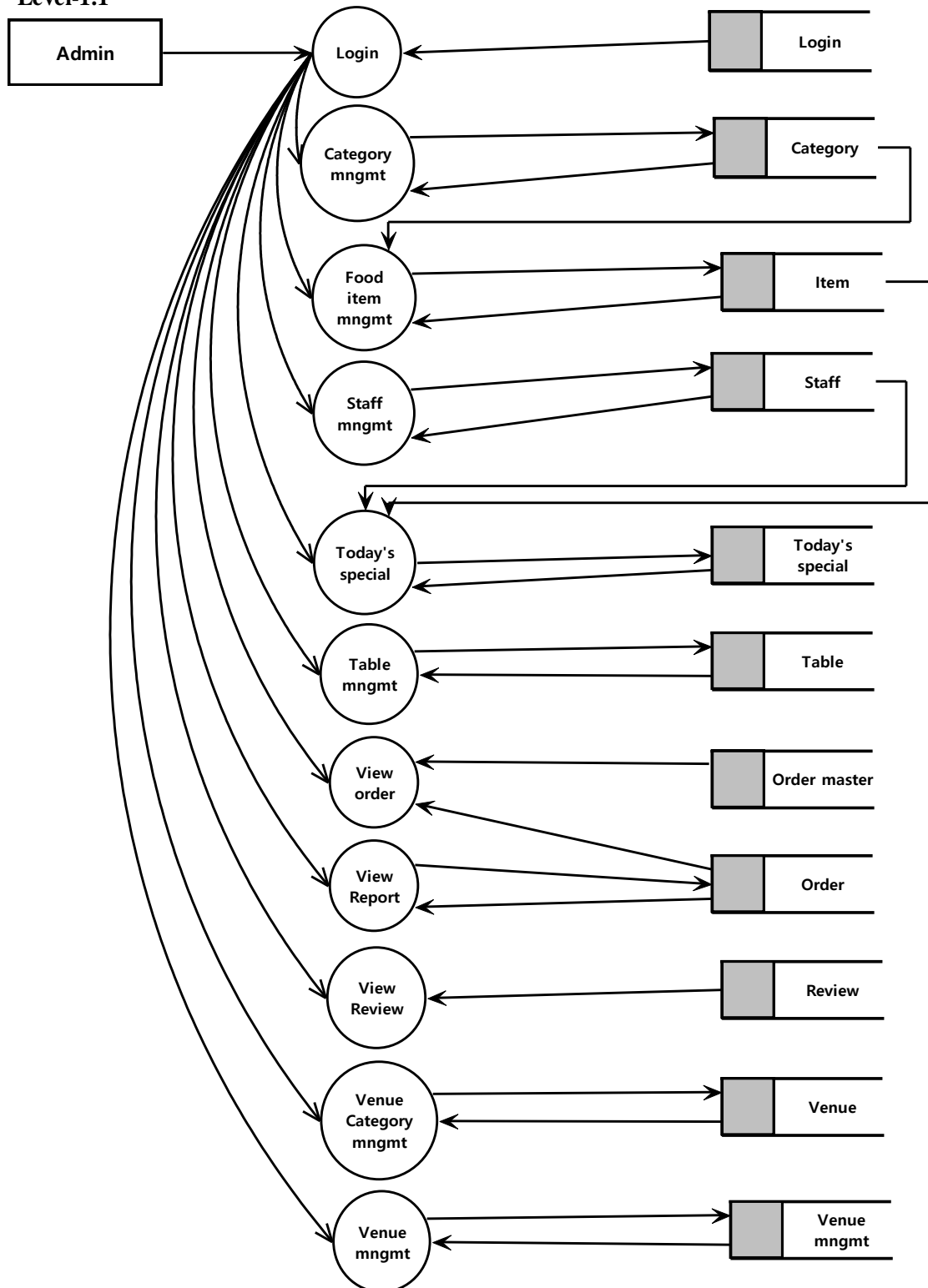
**Four steps are commonly used to construct a DFD:**

- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and left to right.
- When a process is exploded into lower level details they are numbered.
- The names of data stores, sources and destinations are written in Capital letters.

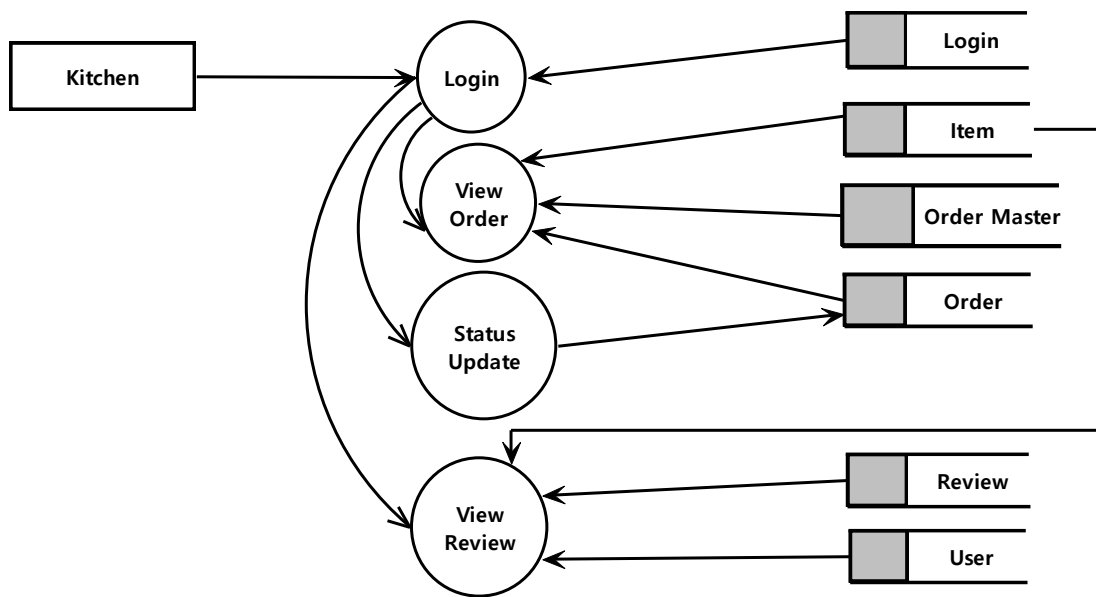
#### DFD Level-1



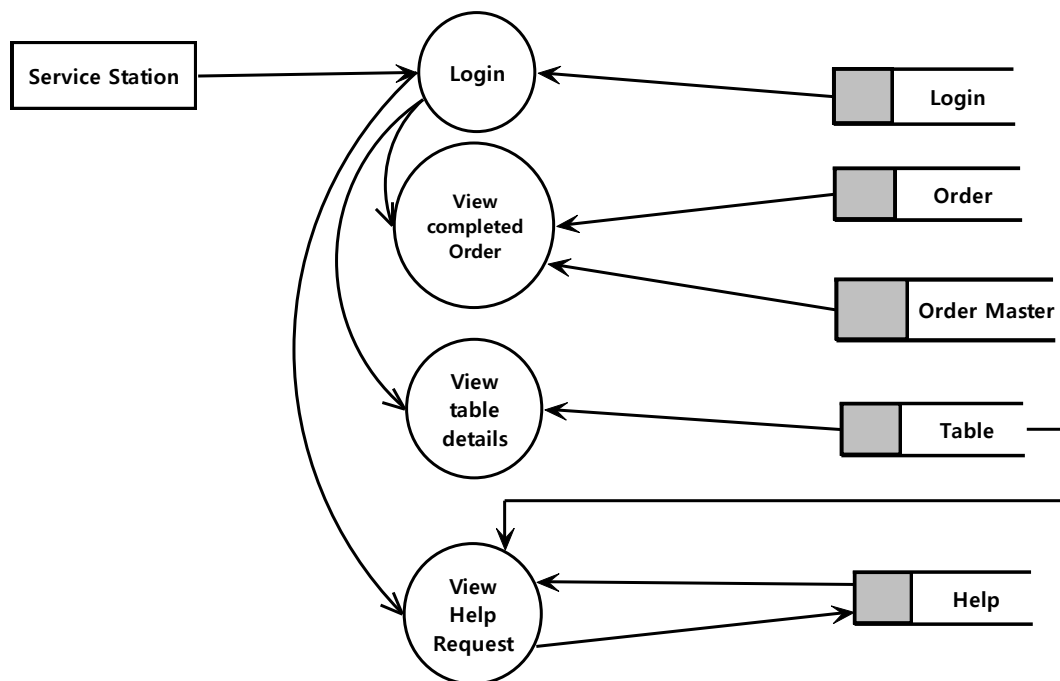
### Level-1.1



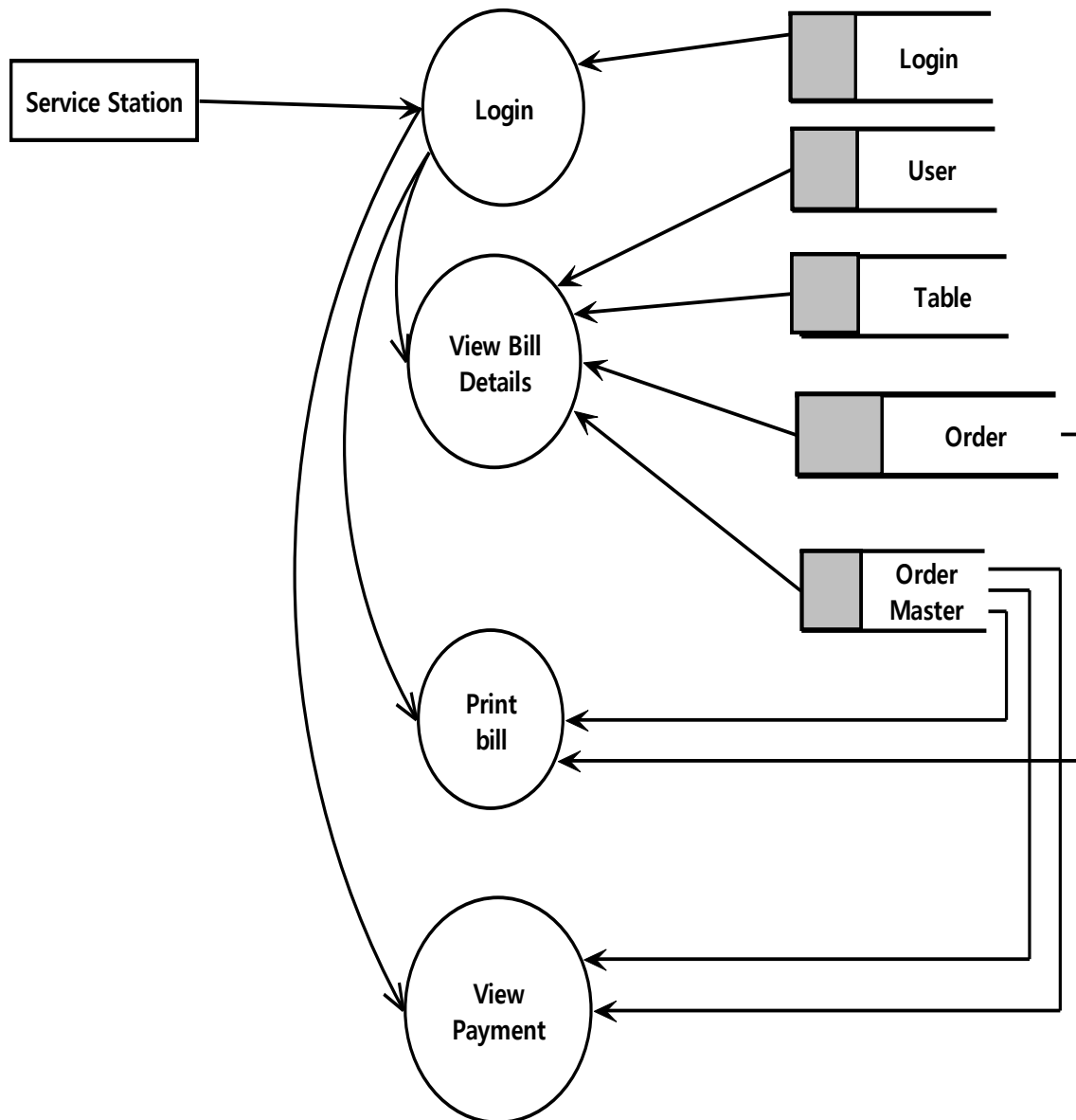
### Level-1.2



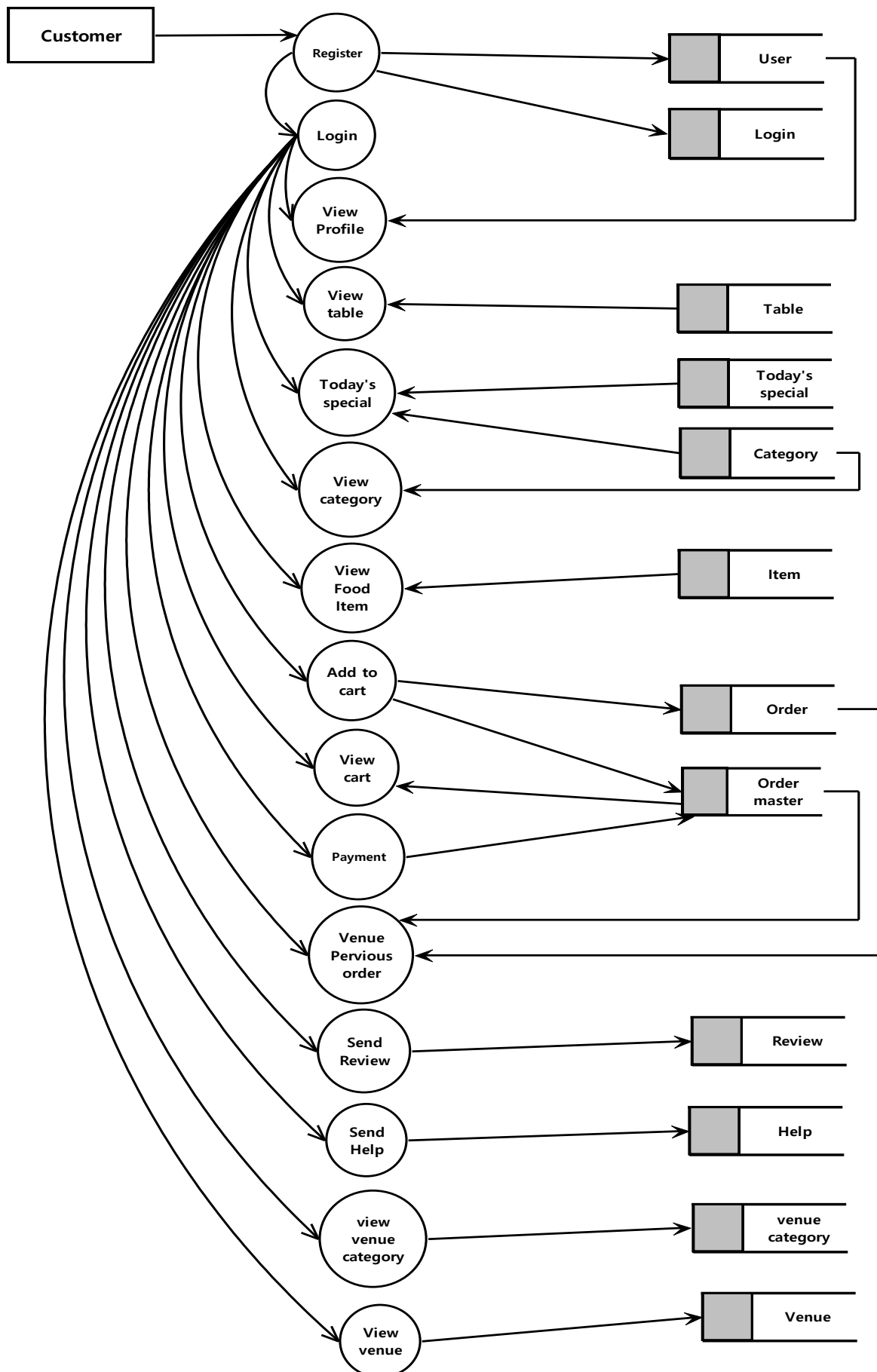
### Level-1.3



## Level-1.4



# Level-1.5





### 3.6 ER Diagram

An ER diagram is a diagram that helps to design databases in an efficient way. It is a data model for describing the data or information. It is a visual representation of data that describes how data is related to each other. The main components of ER models are entities, attributes and the relationships that can exist among them.

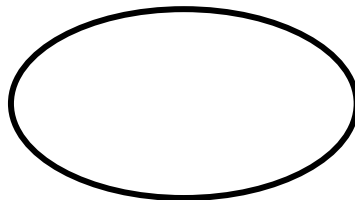
#### Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.



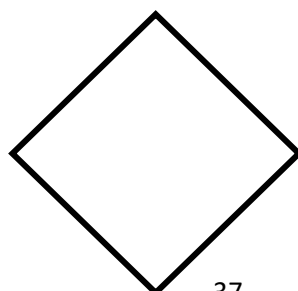
#### Attribute

Attributes are properties of entities. Attributes are represented by means of eclipses. Every eclipse represents one attribute and is directly connected to its entity (rectangle).

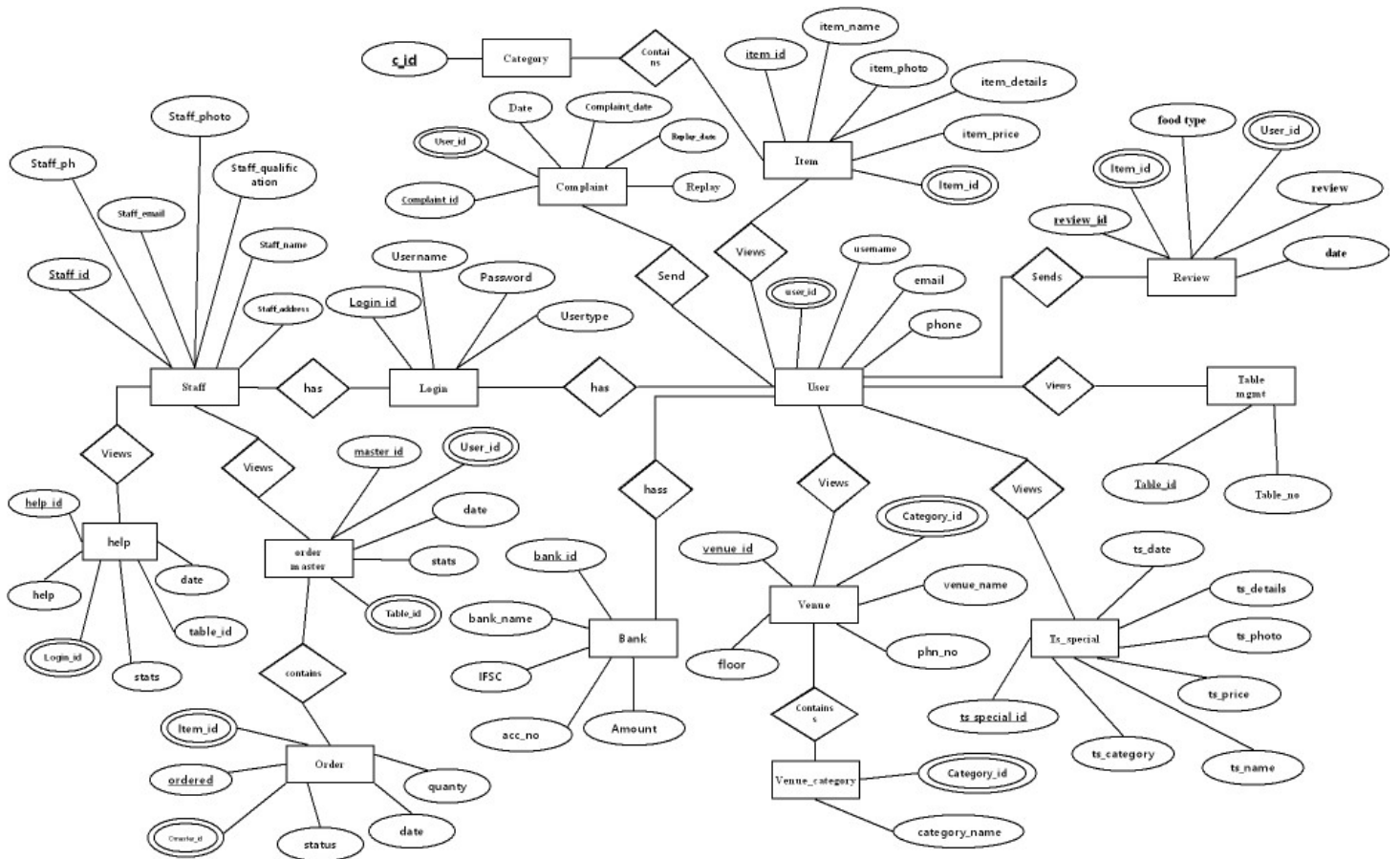


#### Relationship

Relationships are represented by diamond shaped box. Name of the relationship is written in the diamond box. All entities (rectangles), participating in relationship, are connected to it by a line.



## Architectural design



# 4.CODING

## 4.1 INPUT INTERFACE

Input design is a part of overall system design, which requires very careful attention. If data going into the system is correct, then the processing and output will magnify these errors. Thus, the designer has several clear objectives in the different stages of input design.

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that input is acceptable to and understand by the user.

## 4.2 OUTPUT INTERFACE

At the beginning of the output design various types of outputs such as external, internal, operational and interactive and turn around are defined. Then the format, content, location, frequency, volume and sequence of the outputs are specified. The content of the output must be defined in detail. The system analysis has two specific objectives at this stage.

- To interpret and communicate the results of the computer part of a system to the users in a form, which they can understand, and which meets their requirements.
- To communicate the output design specifications to programmers in a way in which it is unambiguous, comprehensive and capable of being translated into a programming language.

## 4.3 SOFTWARE DESCRIPTION

### 4.3.1 HTML

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML

describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML

specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

HTML files are written in ASCII text, so the user can use any text editor to create his/her web page, though a browser of one sort or another is necessary to view the web page. HTML is case insensitive with its language commands. The characters within the document, however, are case sensitive. The language consists of various "tags" which are known as elements. These allow the browser to understand (and put into the desired/specified format) the layout, background, headings, titles, lists, text and/or graphics on the page. The elements are classified according to their function in the HTML document. There are head elements and body elements. The head elements identify properties of the entire document, while body elements actually mark text as content and show a change in the appearance in one way or another. Most elements have a beginning and an ending which encompass the text the user wishes to mark with the tag. All HTML documents must begin with the element and end with the element. Some of the other elements which may be used are tags to create lists-- both ordered lists as well as unordered lists. The user may also create larger or smaller, bolder, italicized, or underlined text. Attributes may be used along with the elements. These perform functions such as placement of text, indication of the source files of images, and identification of links to the document or part of the document.

### **4.3.2 CSS**

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages, user interfaces for web applications, and user interfaces for many mobile applications. CSS is designed primarily to

enable the separation of document content from document presentation, including aspects such as the layout, colours, and fonts.

### **Advantages of CSS**

- CSS saves time – you can write CSS once and then reuse same sheet in multiple HTML pages.

You can define a style for each HTML element and apply it to as many Web pages as you want.

- Pages load faster – If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it

to all the occurrences of that tag. So less code means faster download times.

- Easy maintenance – To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- Superior styles to HTML – CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- Multiple Device Compatibility – Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- Global web standards – Now HTML attributes are being deprecated and it is being recommended to use CSS. So it is a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.
- Offline Browsing – CSS can store web applications locally with the help of an offline cache.

Using of this, we can view offline websites. The cache also ensures faster loading and better overall performance of the website.

- Platform Independence – The Script offer consistent platform independence and can support latest browsers as well.

### 4.3.3 JAVASCRIPT

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript.

The General-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

Advantages of JavaScript:

- Less server interaction – you can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.
- Immediate feedback to the visitors – They don't have to wait for a page reload to see if they have forgotten to enter something.
- Increased interactivity – you can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.
- Richer interfaces – you can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

### 4.3.4 MySQL

MySQL is an open-source relational database management system (RDBMS). MySQL is released under an open-source license. So you have nothing to pay to use it. MySQL is a very powerful program in its own right.

It handles a large subset of the functionality of the most expensive and powerful database packages. It uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.

It works very quickly and works well even with large data sets. It is very friendly to PHP, the most appreciated language for web development. It supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this

(if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB). It is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

### **Major features as available in MySQL 5.6**

- A broad subset of ANSI SQL 99, as well as extensions.
- Cross-platform support.
- Stored procedures, using a procedural language that closely adheres to SQL/PSM.
- Triggers.
- Cursors.
- Updatable views.
- Online DDL when using the InnoDB Storage Engine.
- Information schema.
- Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
- A set of SQL Mode options to control runtime behaviour, including a strict mode to better adhere to SQL standards.
  
- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine.
- Transactions with savepoints when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
- ACID compliance when using InnoDB and NDB Cluster Storage Engines.
- SSL support → Query caching → Sub-SELECTs (i.e. nested SELECTs) .
- Built-in replication support (i.e., master-master replication and master-slave replication) with one master per slave, many slaves per master.
- Multi-master replication is provided in MySQL Cluster, and multimaster support can be added to unclustered configurations using Galera Cluster.
- Full-text indexing and searching.
- Embedded database library.
- Unicode support.
- Partitioned tables with pruning of partitions in optimizer.
- Shared-nothing clustering through MySQL Cluster.

- Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.
- Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.
- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

## **Advantages**

MySQL database server has lots of advantages over its competitors. Some of these advantages have been explained below.

- Open Source and Cost Effective:

The best thing about MySQL server is that this is open source and it has a free version as well.

By open source software, we mean that the code of the software is available and anyone can tailor it according to his requirement. Companies prefer MySQL because they don't have to pay anything for this excellent product.

- Portability:

MySQL is cross platform database server. MySQL can be run on a variety of platforms including Windows, OS2, Linux and Solaris. Portability of MySQL server makes it suitable for applications that target multiple platforms particularly web application. MySQL contains API for almost all the major programming languages and can be easily integrated with the languages like PHP, C++, Perl, C, Python and ruby. In fact, MySQL is a part of the famous LAMP (Linux Apache MySQL PHP) server stack which is used worldwide for web application development.

- Seamless Connectivity:

Various secure and seamless connection mechanisms are available in order to connect with MySQL server. These connections include named pipes, TCP/IP sockets and UNIX Sockets.

- Rapid Development and Continuous Updates:

Being an open source product, MySQL has a very large developer community which releases regular patches and updates for MySQL. Several database templates have been developed which can be readily used and modified resulting in rapid application development.



➤ Security:

MySQL server databases are extremely secure and all the data access scenarios are protected via password and good thing about these passwords is that they are stored in encrypted form and it is not easy to break these advanced and complex encryption algorithms.

#### 4.3.5 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

#### Major features of python

- Easy to code
- Free and Open Source
- Object-Oriented Language
- GUI Programming Support

- High-Level Language
- Extensible feature
- Python is **Portable** language
- Python is Integrated language

### **Advantages**

- Extensive support libraries
- Integration feature
- Improved productivity
- Easy to learn and write
- Vast library support
- Free and open source

### **4.3.6 Flask**

Flask is a web application framework written in Python. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Pocco. Flask is based on the Werkzeug WSGI toolkit and the Jinja2 template engine both are Pocco projects.

It is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier. It gives developers flexibility and is a more accessible framework for new developers since you can build a web application quickly using only a single Python file. Flask is also extensible and doesn't force a particular directory structure or require complicated boilerplate code before getting started.

## **5.CODING PAGES**

## 5.1 Admin Page

```
@app.route('/')
def login():
    return render_template('login_index.html')

@app.route("/login_post", methods=['post'])
def login_post():
    username=request.form['textfield']
    password=request.form['textfield2']
    db=Db()
    res=db.selectOne("select * from login where username='"+username+"' and
password='"+password+"'")
    if res is not None:
        if res['usertype']=='admin':
            session['lg']='lin'
            return redirect("/admin_home")
        if res['usertype']=='kitchen':
            session['lg'] = 'lin'
            return redirect("/kitchen_home")
        if res['usertype']=='staff':
            session['lg'] = 'lin'
            return redirect("/service_home")
        if res['usertype']=='cashier':
            session['lg'] = 'lin'
            return redirect("/cashier_home")
        else:
            return "invalid"

    else :
        return "invalid"

@app.route('/admin_home')
def admin_home():
    if session['lg']=='lin':
        return render_template('admin/admin_index.html')
    else:
        return redirect('/')

@app.route('/add_venue')
def add_venue():
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from venue_category_mngt")
```

```

        return render_template('admin/add_venue.html',data=res)
    else:
        return redirect('/')

@app.route('/add_venue_post', methods=['post'])
def add_venue_post():
    if session['lg'] == 'lin':
        category=request.form['select']
        floor = request.form['textfield']
        venue_name=request.form['textfield2']
        phonenum=request.form['textfield3']
        db=Db()
        db.insert("insert into venue_mngmnt VALUES
(",str(category)+",",venue_name+",",phonenum+",",floor+",")")
        return '<script>alert("added succesfully");window.location="/add_venue"</script>'
    else:
        return redirect('/')

@app.route('/add_venue_ctgry')
def add_venue_ctgry():
    if session['lg'] == 'lin':
        return render_template('admin/add_venue_ctgry.html')
    else:
        return redirect('/')

@app.route("/add_venue_ctgry_post", methods=['post'])
def add_venue_ctgry_post():
    if session['lg'] == 'lin':
        avenue_category=request.form['textfield']
        db=Db()
        db.insert("insert into venue_category_mngt values('"+avenue_category+"',)")
        return '<script>alert("added succesfully");window.location="/add_venue_ctgry"</script>'
    else:
        return redirect('/')

@app.route('/view_review_admin')
def view_review_admin():
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from review,item where review.item_id=item.item_id")
        return render_template('admin/review.html',data=res)
    else:
        return redirect('/')

@app.route('/additem')

```

```

def additem():
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from category")
        return render_template('admin/additem.html',data=res)
    else:
        return redirect('/')
@app.route("/additem_post", methods=['post'])
def additem_post():
    if session['lg'] == 'lin':
        itemname=request.form['textfield2']
        cid=request.form['select']
        details=request.form['textarea']
        photo=request.files['fileField']
        price=request.form['textfield4']
        dt = time.strftime("%Y%m%d_%H%M%S")
        photo.save(static_path + "item\\" + dt + ".jpg")
        path = "/static/item/" + dt + ".jpg"
        db=Db()
        db.insert("insert into item values ('"+cid+"','"+itemname+"','"+
details+"','"+path+"','"+price+"')")
        return '<script>alert("added succesfully");window.location="/additem"</script>'
    else:
        return redirect('/')

@app.route('/addstaff')
def addstaff():
    if session['lg'] == 'lin':
        return render_template('admin/addstaff.html')
    else:
        return redirect('/')
@app.route("/addstaff_post", methods=['post'])
def addstaff_post():
    if session['lg'] == 'lin':
        staffname=request.form['textfield']
        staffhouse=request.form['textfield5']
        staffplace=request.form['textfield7']
        staffpost=request.form['textfield8']
        staffdist=request.form['textfield9']
        pin=request.form['textfield2']
        qualification=request.form['textfield3']
        photo=request.files['fileField']
        email=request.form['textfield4']
        phnumber=request.form['textfield6']
        dt = time.strftime("%Y%m%d_%H%M%S")
        photo.save(static_path + "staff\\" + dt + ".jpg")
        path = "/static/staff/" + dt + ".jpg"
        password=random.randint(0000,9999)
        db=Db()
        qry=db.insert("insert into login VALUES ('"+email+"','"+str(password)+"','staff')")
        db.insert("insert into staff

```

```

values(""+str(qry)+"", ""+staffname+"", ""+staffhouse+"", ""+staffplace+"", ""+staffpost+"", ""+staffdist+"",
""+pin+"", ""+qualification+"", ""+path+"", ""+email+"", ""+phnumber+"")
    return '<script>alert("added succesfully");window.location="/addstaff"</script>'
    else:
        return redirect('/')
@app.route('/addtable')
def addtable():
    if session['lg'] == 'lin':
        return render_template('admin/addtable.html')
    else:
        return redirect('/')
@app.route("/addtable_post", methods=['post'])
def addtable_post():
    if session['lg'] == 'lin':
        tableno=request.form['textfield']
        db=Db()
        db.insert("insert into table_mgmnt(table_no) values('"+tableno+"")")
        return '<script>alert("added succesfully");window.location="/addtable"</script>'
    else:
        return redirect('/')
@app.route('/addtsspecial')
def addtsspecial():
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from category")
        return render_template('admin/addtsspecial.html', data=res)
    else:
        return redirect('/')
@app.route("/addtsspecial_post", methods=['post'])
def addtsspecial_post():
    if session['lg'] == 'lin':
        ctgryname=request.form['select']
        tsname=request.form['textfield2']
        tsdetail=request.form['textarea']
        tsphoto=request.files['fileField']
        tsprice=request.form['textfield3']
        tsdate=request.form['textfield4']
        dt=time.strftime("%Y%m%d_%H%M%S")
        tsphoto.save(static_path + "special_item\\" + dt + ".jpg")
        path="/static/special_item/" + dt + ".jpg"
        db=Db()
        db.insert("insert into ts_special(ts_category,ts_name,ts_details,ts_photo,ts_price,ts_date)
values('"+ctgryname+"', '"+tsname+"', '"+tsdetail+"', '"+path+"', '"+tsprice+"', '"+tsdate+"')")
        return '<script>alert("added succesfully");window.location="/addtsspecial"</script>'
    else:
        return redirect('/')

@app.route('/add_category')
def add_category():
    if session['lg'] == 'lin':

```

```

        return render_template('admin/add_category.html')
    else:
        return redirect('/')
@app.route("/addcategory_post", methods=['post'])
def addcategory_post():
    if session['lg'] == 'lin':
        category=request.form['textfield']
        db=Db()
        qry=db.selectOne("select * from category where category='"+category+"'")
        if qry is not None:
            return '<script>alert("Already added");window.location="/add_category"</script>'
        db.insert("insert into category values('"+category+"")")
        return '<script>alert("added succesfully");window.location="/add_category"</script>'
    else:
        return redirect('/')

@app.route('/edit_venue/<vid>')
def edit_venue(vid):
    if session['lg'] == 'lin':
        db=Db()
        res=db.selectOne("select * from venue_mngmnt where venue_id='"+vid+"'")
        res1=db.select("select * from venue_category_mngt")
        return render_template('admin/edit_venue.html',data=res,data1=res1)
    else:
        return redirect('/')

@app.route("/editvenue_post", methods=['post'])
def editvenue_post():
    if session['lg'] == 'lin':
        evcategory=request.form['select']
        evid=request.form['hid']
        floor=request.form['textfield']
        venue_name=request.form['textfield2']
        phonenumber=request.form['textfield3']
        db=Db()
        db.update("update venue_mngmnt set
category_id='"+evcategory+"',venue_name='"+venue_name+"',phon_no='"+phonenumber+"',floor=
 '"+floor+" where venue_id='"+evid+"'")
        return redirect("/view_venue")
    else:
        return redirect('/')

@app.route('/edit_venue_ctgry/<evid>')
def edit_venue_ctgry(evid):
    if session['lg'] == 'lin':
        db=Db()
        res=db.selectOne("select * from venue_category_mngt where category_id='"+evid+"'")
        return render_template('admin/edit_venue_ctgry.html',data=res)

```



```

else:
    return redirect('/')
@app.route("/edit_venue_ctgry_post", methods=['post'])
def edit_venue_ctgry_post():
    if session['lg'] == 'lin':
        evenuecategory=request.form['textfield']
        evid=request.form['hid']
        db=Db()
        db.update("update venue_category_mngt set category_name='"+evenuecategory+" where
category_id='"+evid+"")
        return redirect("/view_venue_ctgry")
    else:
        return redirect('/')

```

```

@app.route('/Editcategory/<id>')
def Editcategory(id):
    if session['lg'] == 'lin':
        db=Db()
        res=db.selectOne("select * from category where c_id='"+id+"")
        return render_template('admin/Editcategory.html', data=res)
    else:
        return redirect('/')
@app.route("/editcategory_post", methods=['post'])
def editcategory_post():
    if session['lg'] == 'lin':
        ecategory=request.form['textfield']
        ecid=request.form['hid']
        db=Db()
        db.update("update category set category='"+ecategory+" where c_id='"+ecid+"")
        return redirect("/viewcategory")
    else:
        return redirect('/')
@app.route('/edititem/<itid>')
def edititem(itid):
    if session['lg'] == 'lin':
        db=Db()
        res=db.selectOne("select * from item where item_id='"+itid+"")
        res1=db.select("select * from category")
        return render_template('admin/edititem.html',data=res,data1=res1)
    else:
        return redirect('/')
@app.route("/edititem_post", methods=['post'])
def edititem_post():
    if session['lg']=='lin':
        categoryname=request.form['select']
        itemid=request.form['hid']
        itemname=request.form['textfield2']
        details=request.form['textarea']
        price=request.form['textfield4']

```

```

db = Db()
db.update(
    "update item set item_cid=" + categoryname + ",item_name=" + itemname +
    ",item_details=" + details + ",item_price=" + price + " where item_id=" + itemid + """)

if 'fileField' in request.files:
    photo = request.files['fileField']
    if photo.filename!="":
        dt = time.strftime("%Y%m%d_%H%M%S")
        photo.save(static_path + "item\\" + dt + ".jpg")
        path = "/static/item/" + dt + ".jpg"
        db = Db()
        db.update("update item set item_photo=" + path + " where item_id=" + itemid + "")

    else:
        pass
else:
    pass
return redirect("/viewitem")

else:
    return redirect('/')

@app.route('/review/<it_id>/<type>')
def review(it_id, type):
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from review,user where user.user_id=review.user_id and
item_id="+it_id+"and food_type="+type+"")
        return render_template('admin/review.html',data=res)
    else:
        return redirect('/')
@app.route("/review_post", methods=['post'])
def review_post():
    if session['lg'] == 'lin':
        categoryname=request.form['textfield']
    else:
        return redirect('/')
@app.route('/updatestaff/<sid>')
def updatestaff(sid):
    if session['lg'] == 'lin':
        db=Db()
        res=db.selectOne("select * from staff where staff_id="+sid+"")
        return render_template('admin/updatestaff.html',data=res)
    else:
        return redirect('/')
@app.route("/upatestaff_post", methods=['post'])
def upatestaff_post():
    if session['lg'] == 'lin':
        staffid=request.form['hid']
        staffname=request.form['textfield']

```

```

staffhouse = request.form['textfield5']
staffplace = request.form['textfield7']
staffpost = request.form['textfield8']
staffdist = request.form['textfield9']
pin=request.form['textfield2']
qualification=request.form['textfield3']
email=request.form['textfield4']
phnumber=request.form['textfield6']
db=Db()
db.update("update staff set
staff_name=""+staffname+"",staff_hn=""+staffhouse+"",staff_place=""+staffplace+"",staff_post=""+st
affpost+"",staff_district=""+staffdist+"",staff_pin=""+pin+"",staff_qualification=""+qualification+"",sta
ff_email=""+email+"",staff_phone=""+phnumber+"" where staff_id=""+staffid+"")
    if 'fileField' in request.files:
        photo = request.files['fileField']
        if photo.filename!="":
            dt = time.strftime("%Y%m%d_%H%M%S")
            photo.save(static_path + "item\\" + dt + ".jpg")
            path = "/static/item/" + dt + ".jpg"
            db = Db()
            db.update("update staff set staff_photo=" + photo + " where staff_id="" + staffid + """)

    else:
        pass
else:
    pass

    return redirect("/viewstaff")
else:
    return redirect('/')
@app.route('/updatetsspecial/<tsid>')
def updatetsspecial(tsid):
    if session['lg'] == 'lin':
        db=Db()
        res=db.selectOne("select * from ts_special where ts_special_id=""+tsid+""")
        res1=db.select("select * from category")
        return render_template('admin/updatetsspecial.html', data=res,data1=res1)
    else:
        return redirect('/')
@app.route("/updatetsspecial_post", methods=['post'])
def updatetsspecial_post():
    if session['lg'] == 'lin':
        tsspecialid=request.form['hid']
        ctgryname=request.form['select']
        tsname=request.form['textfield2']
        tsdetail=request.form['textarea']
        tsprice=request.form['textfield3']
        tsdate=request.form['textfield4']
        db = Db()

```

```

        db.update("update ts_special set
ts_category='"+ctgryname+"',ts_name='"+tsname+"',ts_details='"+tsdetail+"',ts_price='"+tsprice+"
,ts_date='"+tsdate+"' where ts_special_id='"+tsspecialid+"'")
        if 'fileField' in request.files:
            photo = request.files['fileField']
            if photo.filename != "":
                dt = time.strftime("%Y%m%d_%H%M%S")
                photo.save(static_path + "item\\" + dt + ".jpg")
                path = "/static/item/" + dt + ".jpg"
                db = Db()
                db.update("update ts_special set ts_photo='"+photo+"' where ts_special_id='"+
+tsspecialid+"'")

            else:
                pass
        else:
            pass

        return redirect("/viewtsspecial")
    else:
        return redirect('/')

@app.route('/view_venue')
def view_venue():
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from venue_mngmnt, venue_category_mngt where
venue_category_mngt.category_id=venue_mngmnt.category_id")
        return render_template('admin/view_venue.html',data=res)
    else:
        return redirect('/')

@app.route("/delete_venue/<vid>")
def delete_venue(vid):
    if session['lg'] == 'lin':
        db=Db()
        db.delete("delete from venue_mngmnt where venue_id='"+vid+"'")
        return redirect("/view_venue")
    else:
        return redirect('/')

@app.route('/view_venue_ctgry')
def view_venue_ctgry():
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from venue_category_mngt")
        return render_template('admin/view_venue_ctgry.html', data=res)
    else:
        return redirect('/')

```

```

@app.route("/delete_venue_category/<id>")
def delete_venue_category(id):
    if session['lg'] == 'lin':
        db=Db()
        db.delete("delete from venue_category_mngt where category_id='"+id+"'")
        return redirect("/view_venue_ctgry")
    else:
        return redirect('/')

@app.route('/viewcategory')
def viewcategory():
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from category")
        return render_template('admin/viewcategory.html',data=res)
    else:
        return redirect('/')

@app.route("/delete_category/<cid>")
def delete_category(cid):
    if session['lg'] == 'lin':
        db=Db()
        db.delete("delete from category where c_id='"+cid+"'")
        return redirect("/viewcategory")

@app.route('/viewitem')
def viewitem():
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from item,category where category.c_id=item.item_cid")
        return render_template('admin/viewitem.html',data=res)
    else:
        return redirect('/')

@app.route("/delete_item/<tid>")
def delete_item(tid):
    if session['lg'] == 'lin':
        db=Db()
        db.delete("delete from item where item_id='"+tid+"'")
        return redirect("/viewitem")
    else:
        return redirect('/')

@app.route('/ViewOrder')
def ViewOrder():
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from order_master,table_mgmnt where table_mgmnt.table_id=order_master.table_id and date=curdate()")
        return render_template('admin/ViewOrder.html',data=res)
    else:
        return redirect('/')

```

```

@app.route('/viewordereditem/<mid>')
def viewordereditem(mid):
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select item.item_name as name, item.item_price as price, order.qty,
`order`.item_type from item,`order` where `order`.item_type='item' and
item.item_id=order.item_id and order.omaster_id='"+mid+"' union(select ts_special.ts_name as
name, ts_special.ts_price as price, order.qty, `order`.item_type from ts_special,`order` where
`order`.item_type='special_item' and ts_special.ts_special_id=order.item_id and
order.omaster_id='"+mid+"'")
        print(res)
        return render_template('admin/viewordereditem.html',data=res)
    else:
        return redirect('/')

```

```

@app.route('/ViewReport', methods=['get', 'post'])
def ViewReport():
    if session['lg'] == 'lin':
        if request.method=="POST":
            t1=request.form['t1']
            t2=request.form['t2']
            db=Db()
            res=db.select("select date, sum(amount) as amount from order_master where date
between '"+t1+"' and '"+t2+"' group by date")
            return render_template('admin/ViewReport.html', data=res)
        else:
            return render_template('admin/ViewReport.html')
    else:
        return redirect('/')

```

```

@app.route('/viewstaff')
def viewstaff():
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from staff")
        return render_template('admin/viewstaff.HTML',data=res)
    else:
        return redirect('/')

```

```

@app.route("/delete_staff/<sid>")
def delete_staff(sid):
    if session['lg'] == 'lin':
        db=Db()
        db.delete("delete from staff where staff_id='"+sid+"'")
        return redirect("/viewstaff")
    else:
        return redirect('/')

```

```

@app.route('/viewtable')

```

```

def viewtable():
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from table_mgmnt")
        return render_template('admin/viewtable.html',data=res)
    else:
        return redirect('/')

@app.route("/delete_table/<tbid>")
def delete_table(tbid):
    if session['lg'] == 'lin':
        db=Db()
        db.delete("delete from table_mgmnt where table_id='"+tbid+"'")
        return redirect("/viewtable")
    else:
        return redirect('/')

@app.route('/viewtsspecial')
def viewtsspecial():
    if session['lg'] == 'lin':
        db=Db()
        res=db.select("select * from ts_special,category where category.c_id=ts_special.ts_category")
        return render_template('admin/viewtsspecial.html',data=res)
    else:
        return redirect('/')

@app.route("/delete_tsspecial/<tsid>")
def delete_tsspecial(tsid):
    if session['lg'] == 'lin':
        db=Db()
        db.delete("delete from ts_special where ts_special_id='"+tsid+"'")
        return redirect("/viewtsspecial")
    else:
        return redirect('/')

```

## **6. SYSTEM TESTING**



## **6.1 TESTING AND EVALUATION**

Testing is a process of executing a program with the intent of finding an error. Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design and coding. Testing includes verifications of the basic logic of each program and verification that the entire system works properly. Testing demonstrates that software functions appear to be working according to specification. In addition, data collected as testing is conducted provides a good indication of software quality as a whole. The debugging process is the most unpredictable part of the testing process. Testing begins at the module level and works towards the integration of the entire computer-based system. Testing and debugging are different activities, but any testing includes a debugging strategy. For software testing, one must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements. No testing is complete without a verification and validation part. The goals of verification and validation activities are to assess and improve the quality of work products generated during the development and modification of the software. There are two types of verification: life cycle verification and formal verification. Life cycle verification is the process of determining the degree to which the products of the given phase of the development cycle fulfill the specification established during the prior process. Formal verification is the rigorous mathematical demonstration that source code conforms to its specifications. Validation is a process of evaluating software at the end of the software development process to determine conformance with the requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The primary objectives, when we test software are the following:

- Testing is a process of exceeding with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.
- A successful test is one uncovers undiscovered errors.

Thus, testing plays a very critical role in determining the reliability and efficiency of the software and hence is very important stage in software development. Tests are to be conducted on the software to evaluate its performance under a number of conditions. Ideally, it should so at the level

of each module and also when all of them are integrated to from the completed system. Software testing is done at different levels.

## **6.2 TESTING STRATEGIES**

A strategy for software testing integrates software test case design method in to a well-planned series of steps that result in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote a customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project progresses.

**The general characteristics of software testing strategies are:**

- Testing begins at the component level and works “outward” toward the integration of the entire computer system.
- Different testing techniques are appropriate at different points in time.

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

A strategy must provide guidance for the practitioner and set of milestones for the manager. Because the step on the test strategy occurs at a time when deadline pressure begins to rise, progress must be measurable and problem must surface as early as possible.

The software team's approach to testing is defining a plan that describes an overall strategy and a procedure that defines specific testing steps and tests that will be conducted. In the proposed system, if the administrator makes any attempt to login to the application without entering his password, then the system will not allow the user to login to the application.

## **6.3 TESTING TECHNIQUES**

The various testing techniques are given below:

### **6.3.1 WHITE BOX TESTING**

White-box testing is also called as glass-box testing, is a test case design method that goes to the control structure of the procedural design to derive

test cases. Using white box testing methods, the software engineer can derive test cases that,

- ✓ Guarantee that all independent paths within a module have been exercised at Least once.
- ✓ Exercise all logical decision on their true and false sides.
- ✓ Execute all loops at their boundaries and within their operational sides.
- ✓ Exercise internal data structure to ensure their validity.

White box testing was successfully conducted on our system. All independent paths within a module have been executed at least once and all logical decisions have been exercised on their true and false sides.

### **6.3.2 BLACK BOX TESTING**

Black-box testing is also called as behavioural testing, focuses on the functional requirement of the software. It is a complimentary approach that is likely uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories.

- ✓ Incorrect or missing functions.
- ✓ Interface errors.
- ✓ Error on data structures or external database access.
- ✓ Behaviour or performance errors.
- ✓ Initialization and termination errors.

Black box testing was successfully conducted on your system. The system was divided into a number of modules and testing was conducted on each module. We have tested the system for incorrect or missing functions, interface and performance errors.

### **6.3.3 UNIT TESTING**

Unit testing comprises the set of tests performed by an individual programmer prior to the integration of the system. Testing removes residual bugs and improves the reliability of the system.

Testing allows the developer to find out the design faults if any, and enable correction if needed. Exhaustive unit testing has to be carried out to ensure the validity of the data. In order to successfully test the entire package, unit testing is carried out. Each module was tested as when it was developed. Thus it proved easier to conduct minute testing operation and correct them then and there.

### **6.3.4 INTEGRATION TESTING**

Bottom-up integration is the traditional strategy used to integrate the component of a software system into a functional whole. Bottom-up integration consists of unit testing, followed by subsystem testing and followed by testing of the entire system. Unit testing has the goal of discovering errors in the individual parts of the system.

Parts are tested in isolation from one another in an artificial environment known as “Test Harness”, which consists of driver programs and data necessary to exercise the modules. Unit testing should be as exhaustive as possible to ensure that each representative case handled by each module has been tested. Unit testing is eased by a system structure that is composed of small loosely coupled modules.

Both control and data interfaces must be tested. Large software system may require several levels of subsystem testing. Lower level subsystems are successively combined to form higher level subsystems. In most software systems, exhaustive testing of subsystem capabilities is not feasible due to the combination complexity of the module interfaces. Therefore, test cases must be carefully chosen to exercise the interfaces in the desired manner.

### **6.3.5 ACCEPTANCE TESTING**

Acceptance testing involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements. It is not unusual for two sets of acceptance test to be run, those developed by the quality group and those developed by the customer.

In addition to the functional and performance tests, stress tests are performed to determine the limitation of the system. For example, a compiler might be tested to determine the effect of the symbol table overflow, or real-time system might be tested to determine the effect of simultaneous arrival of numerous high priority interrupts.

### **6.3.6 OUTPUT TESTING**

Output testing of the proposed system is important since no system could be useful if it does not produce the required output.

The output format on the screen is found to be correct, as the format was designed in the system design phase according to the user needs. For the hard copy also the output comes out as the specified requirements by

the user. Hence output testing doesn't result in any correction on the system.

## **7.SYSTEM IMPLEMENTATION AND DEPLOYMENT**

Implementation is the process of deploying the new system in the operational environment. Proper implementation is essential to provide a reliable system to meet the organizational requirement. There are four types of implementation methods. They are Direct Changeover, Phased Implementation, Parallel Run and Pilot Approach. The most commonly used implementation methods are Pilot Approach and Parallel Run.

The system which is developed as a web application hence the other functions as normal application, as usual some web development technologies are used in the implementation of the project. The language I selected to program this software is PHP. The reason I selected PHP is that is a simple and powerful language that especially developed to create web application.

Technologies used in the development of the software are:

- ✓ Programming language – Python
- ✓ DBMS – MySQL server
- ✓ Development tool – PyCharm

✓ Development platform – Windows 10

The front end is HTML, and CSS and back end is MySQL Server and Python. The system developed on Pycharm in Windows 10 operating system.

## **8. CONCLUSION**

In conclusion, the implementation of a digital menu system offers numerous benefits to both customers and restaurant owners. Customers can enjoy a more interactive and engaging dining experience, with the ability to easily browse menu items, view images, and access additional information. Restaurant owners can benefit from increased efficiency, reduced printing costs, and the ability to easily update their menu offerings. Additionally, a digital menu system can help to streamline the ordering and payment process, resulting in shorter wait times and improved customer satisfaction. Overall, the use of a digital menu system can greatly enhance the dining experience for both customers and restaurant owners, and is a worthwhile investment for any establishment looking to improve their operations and stay ahead in the increasingly competitive food industry.

### **8.1 FUTURE ENHANCEMENT**

As technology continues to evolve, there are many potential future enhancements that could be made to digital menu systems.

One possible enhancement is the integration of augmented reality (AR) technology, which could allow customers to see virtual 3D models of dishes, view nutritional information, and even order their meals directly from the AR menu. Another potential enhancement is the use of artificial intelligence (AI) to personalize the menu for each individual customer based on their preferences, order history, and dietary restrictions.

Another future enhancement could be the integration of blockchain technology, which could provide increased transparency and security in the supply chain. With blockchain, customers could scan a QR code on the menu to view information about where their food was sourced, how it was prepared, and any certifications or quality standards that it meets.

Finally, there is the potential to incorporate voice recognition technology, which could allow customers to place their orders directly with a virtual assistant, further streamlining the ordering process and improving efficiency. With these future enhancements and many more, digital menu systems have the potential to revolutionize the dining experience, making it more interactive, personalized, and enjoyable for customers while also improving the operations and profitability of restaurants.

## **9. REFERENCE**

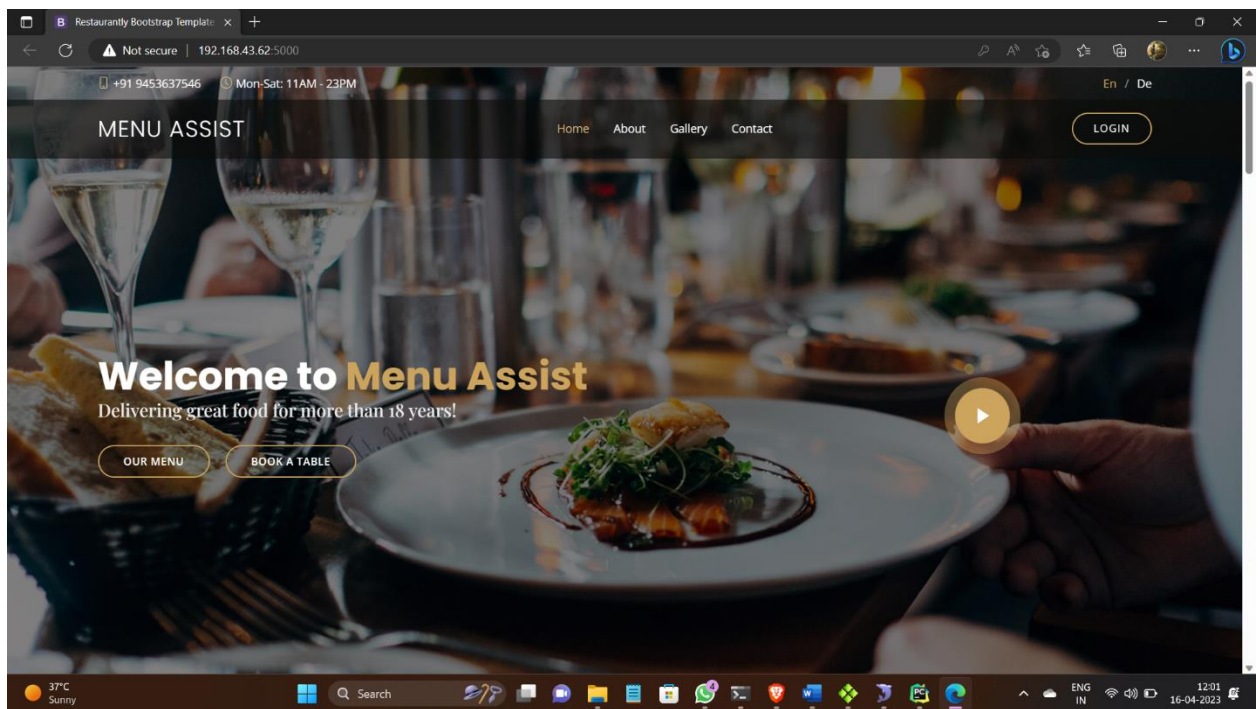
- Google.com
- YouTube

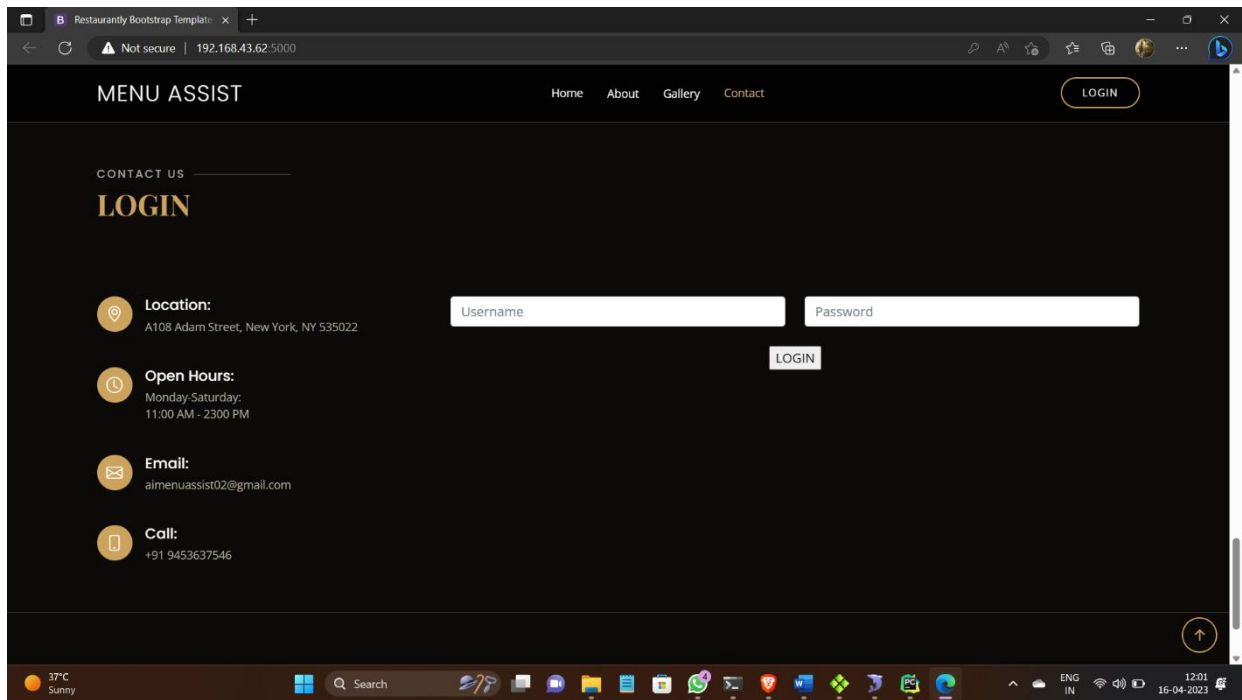


# 10. APPENDIX

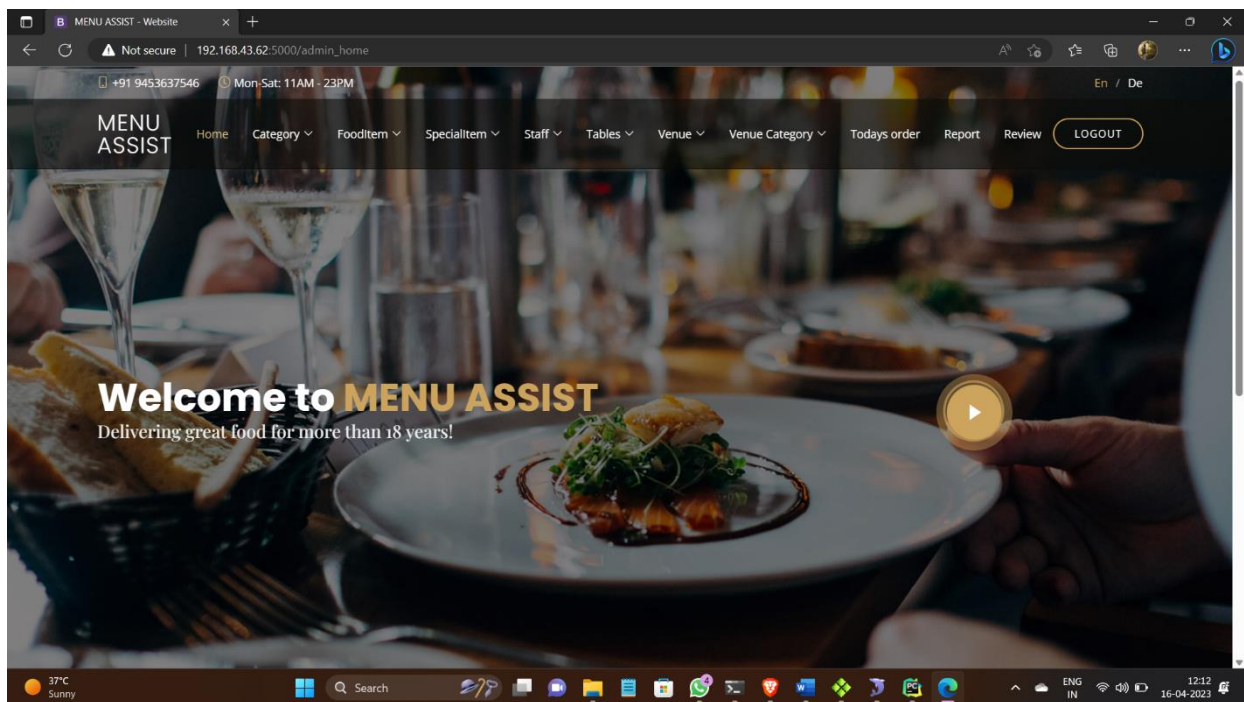
## 10.1 Website

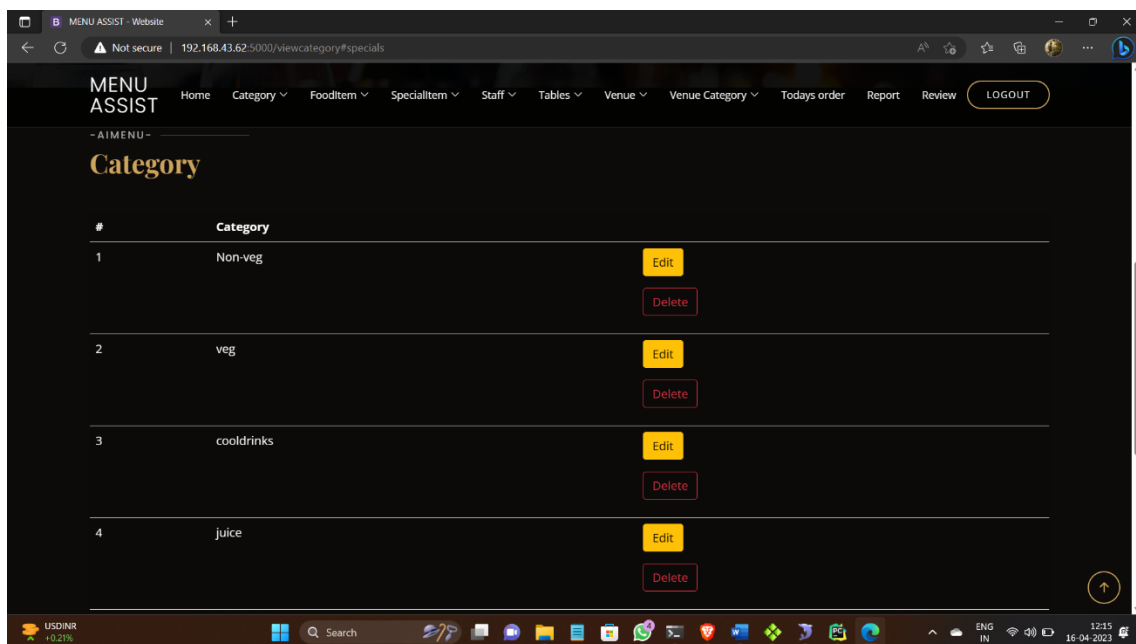
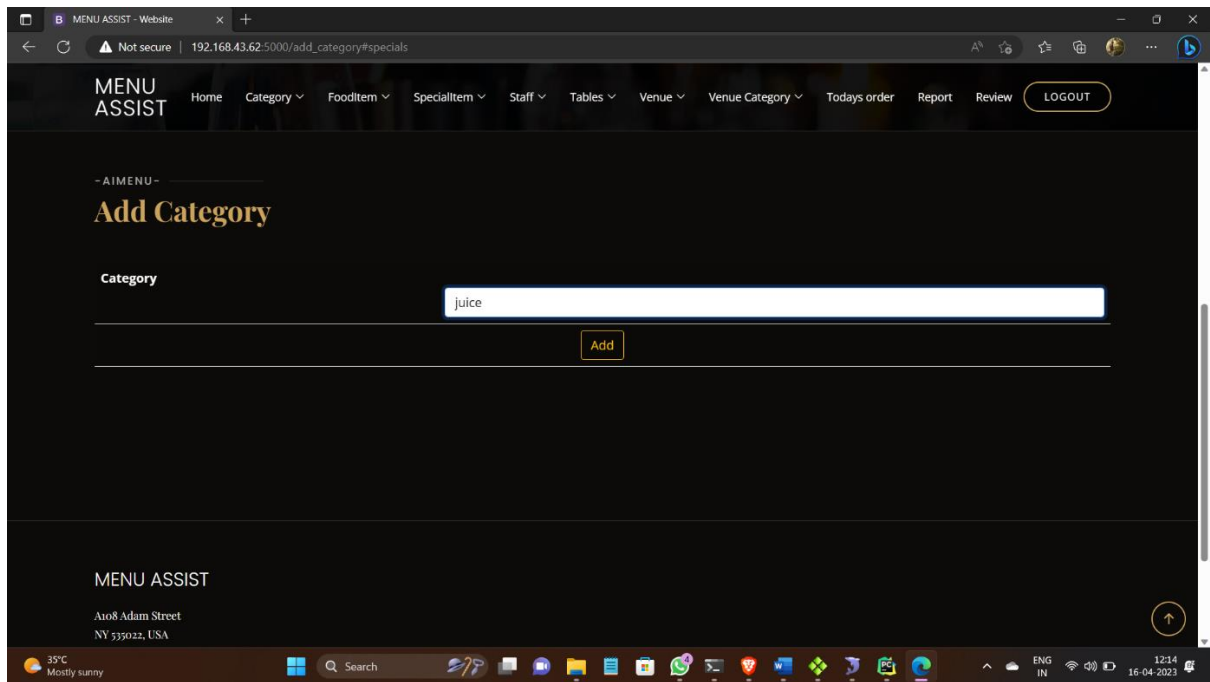
### *10.1.1 Loginpage:*

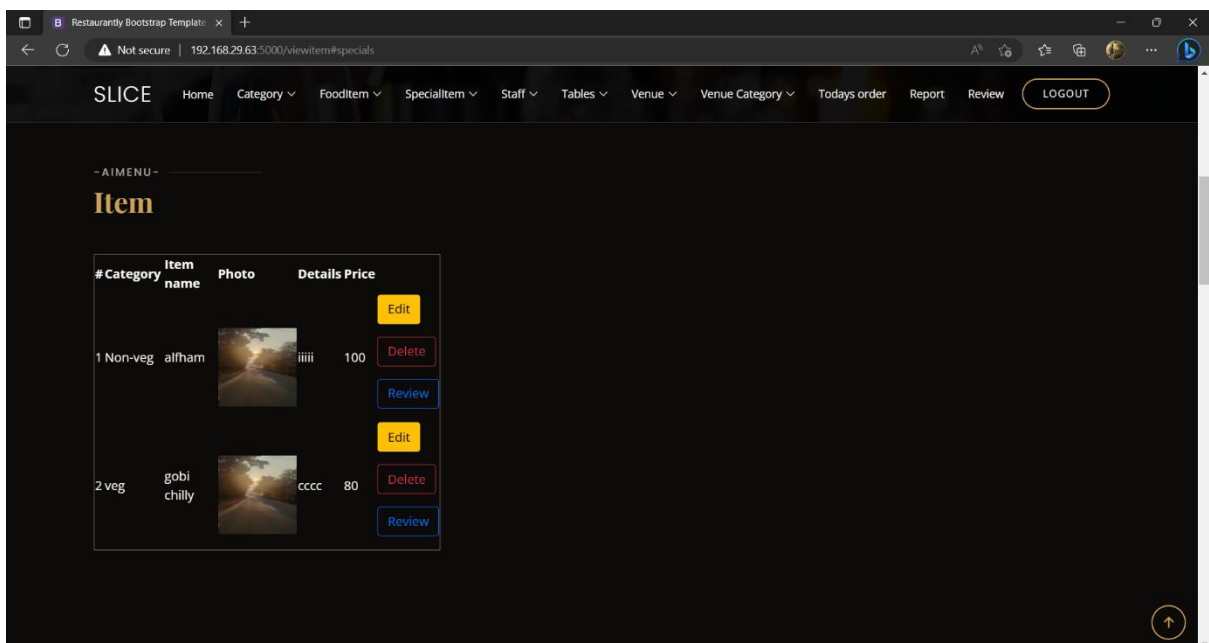
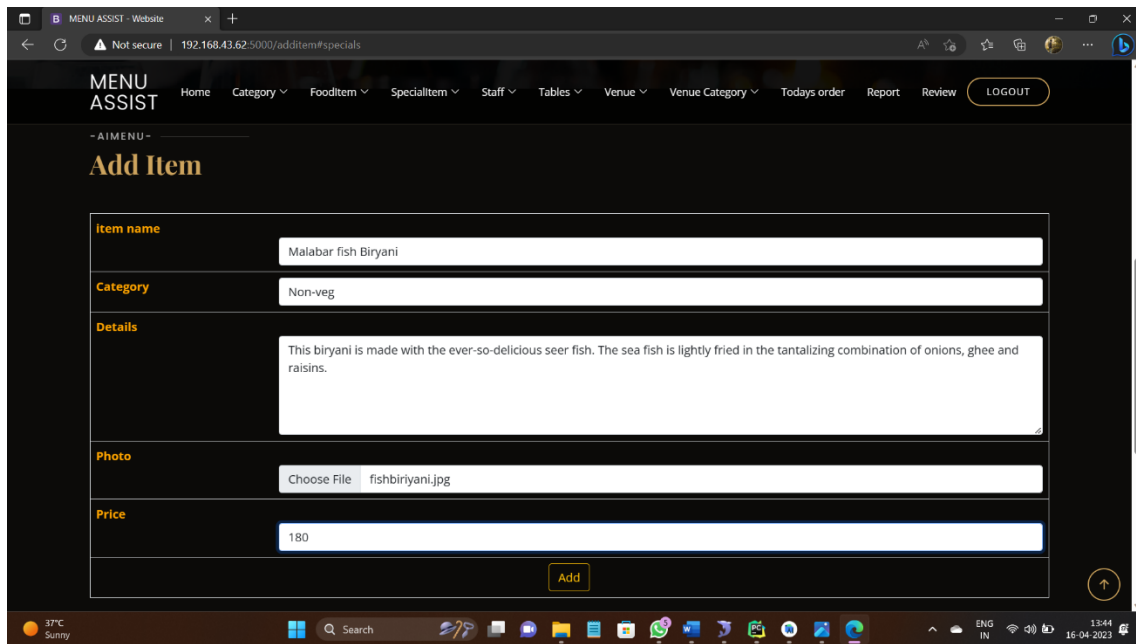




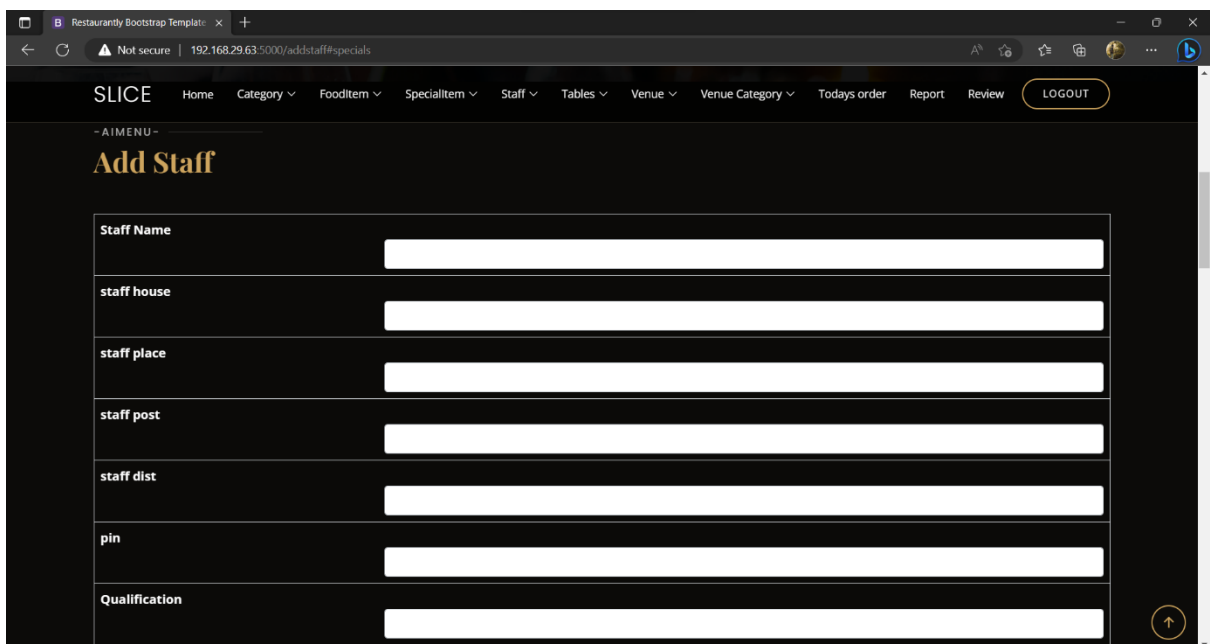
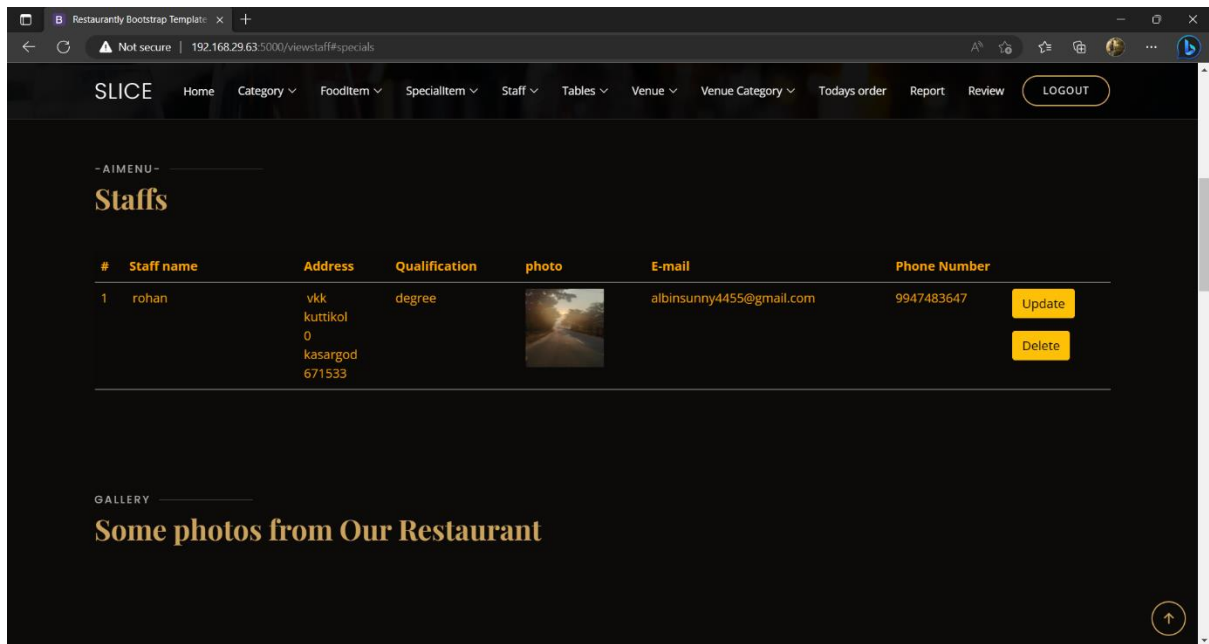
### 10.1.2 AdminPage:











MENU ASSIST - Website

192.168.43.62:5000/ViewReport#specials

MENU ASSIST

[Home](#)
[Category](#)
[FoodItem](#)
[SpecialItem](#)
[Staff](#)
[Tables](#)
[Venue](#)
[Venue Category](#)
[Todays order](#)
[Report](#)
[Review](#)

LOGOUT

- AI MENU -

Report

INCOME REPORT

From

dd-mm-yyyy

To

dd-mm-yyyy

Search

#	DATE	AMOUNT
1	2023-03-21	740.0

MENU ASSIST

A108 Adam Street

NY 535022, USA

Phone: +91 9453637546

35°C Mostly sunny

Search

ENG IN

12:21

16-04-2023

MENU ASSIST - Website

192.168.43.62:5000/view\_review\_admin#specials

MENU ASSIST

[Home](#)
[Category](#)
[FoodItem](#)
[SpecialItem](#)
[Staff](#)
[Tables](#)
[Venue](#)
[Venue Category](#)
[Todays order](#)
[Report](#)
[Review](#)

LOGOUT

- AI MENU -

View Review

#	Date	Review
1	2023-03-18	spicy
2	2023-03-19	good
3	2023-03-19	Need to be improved
4	2023-03-21	good
5	2023-03-21	good
6	2023-03-21	good
7	2023-03-21	sss
8	2023-03-21	fagaga
9	2023-03-21	potta food

MENU ASSIST

A108 Adam Street

NY 535022, USA

Phone: +91 9453637546

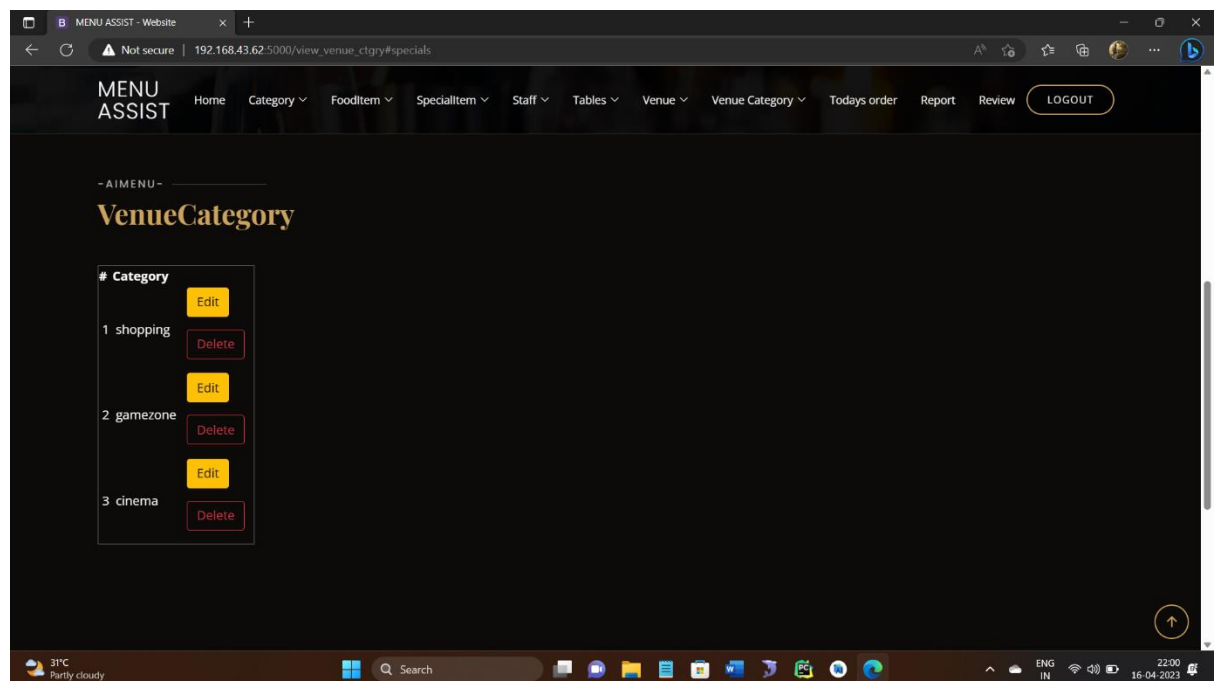
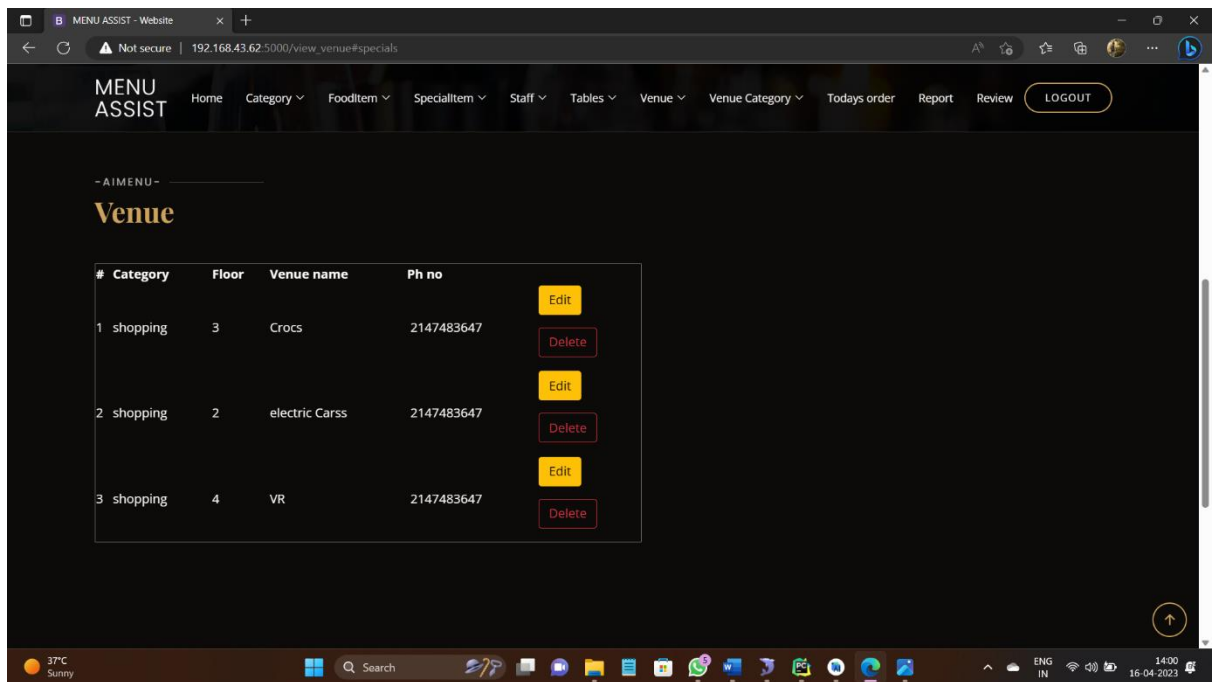
35°C Mostly sunny

Search

ENG IN

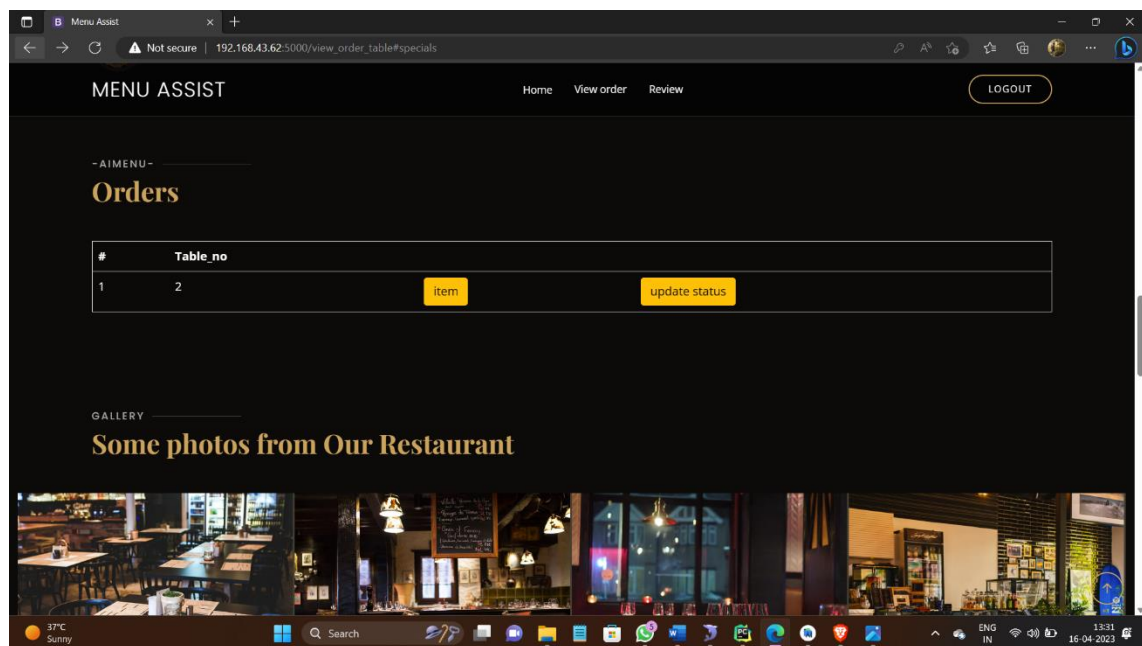
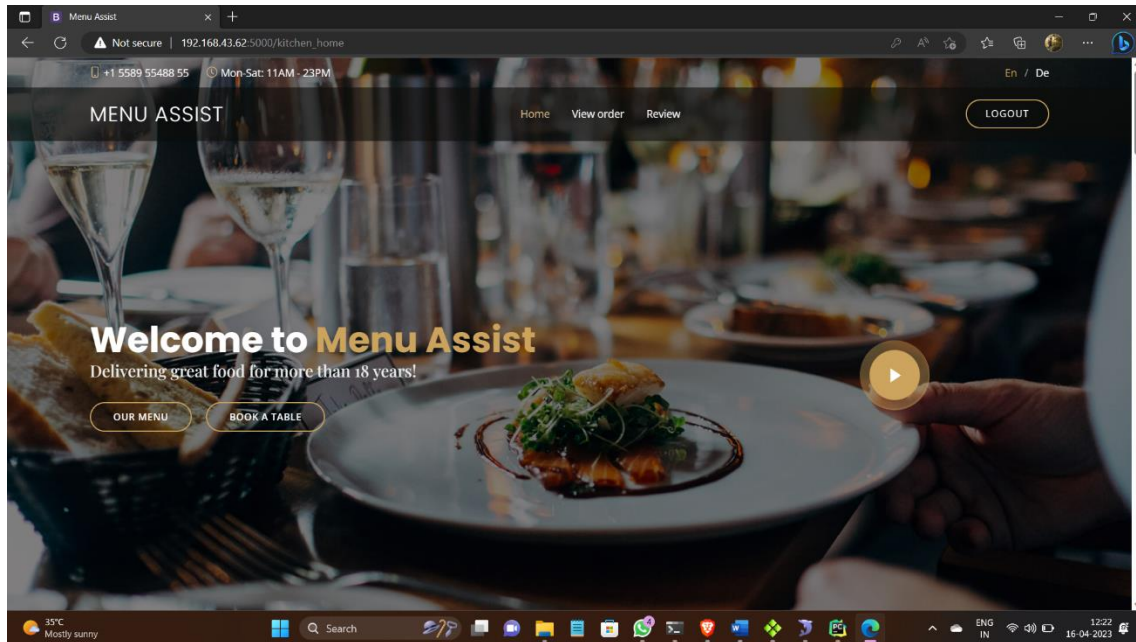
12:21

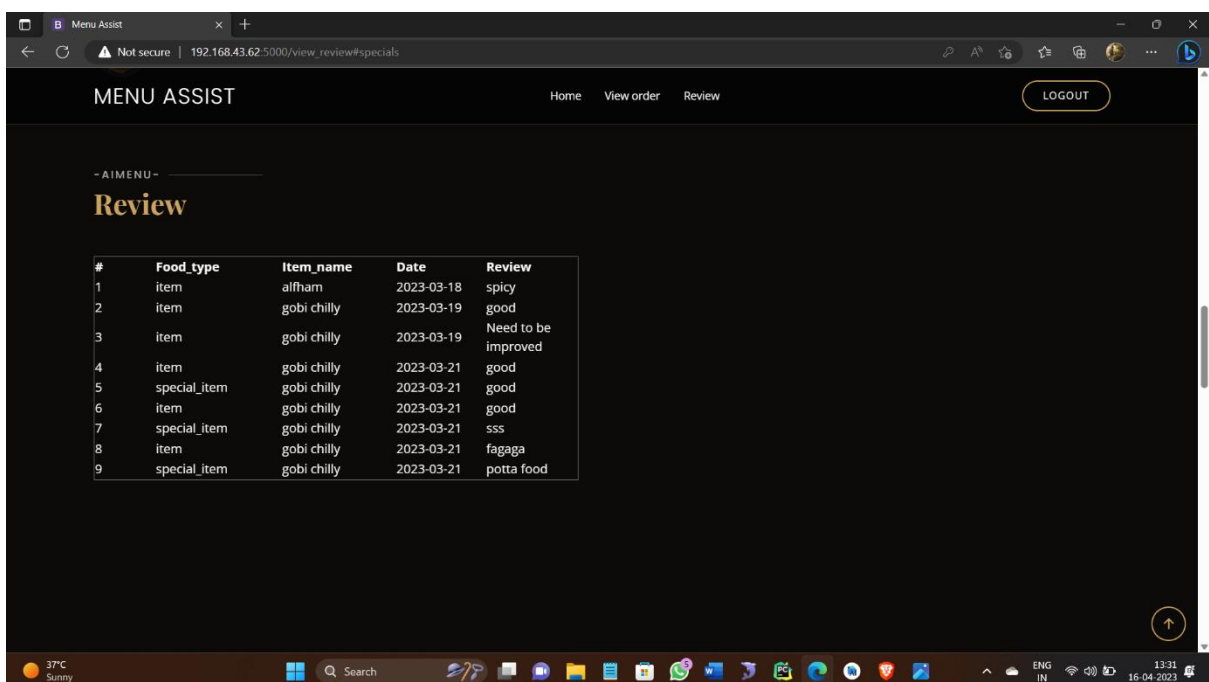
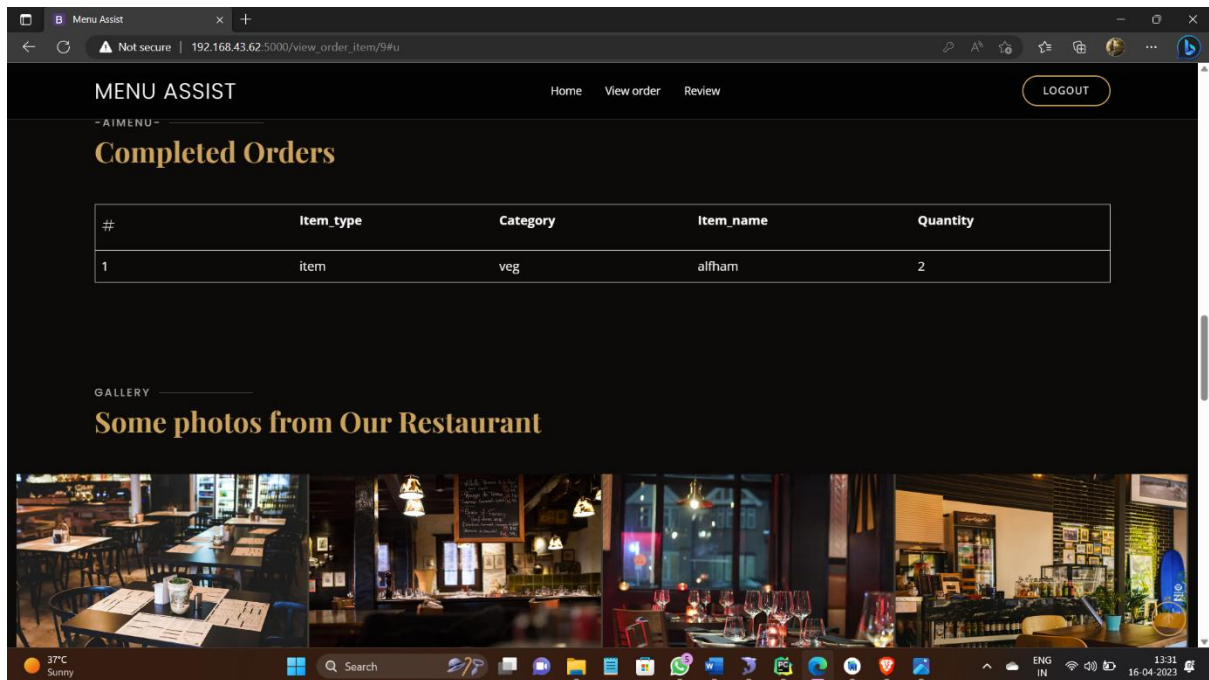
16-04-2023



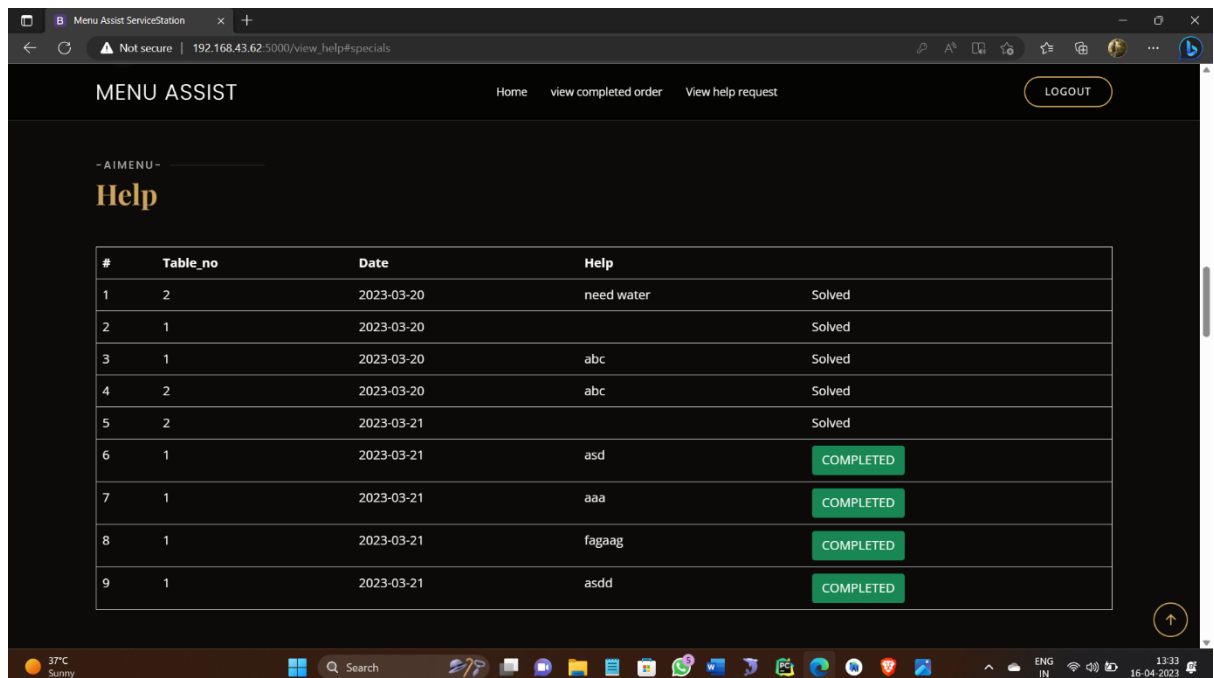
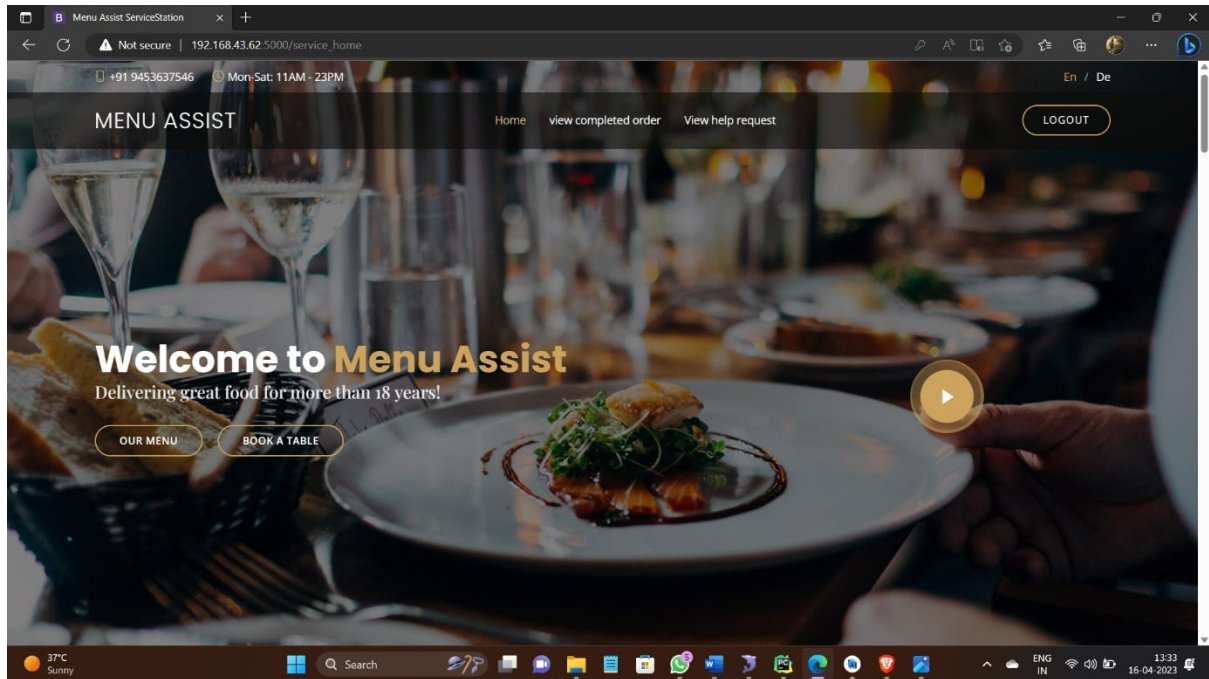


### 10.1.3 kitchen Page:

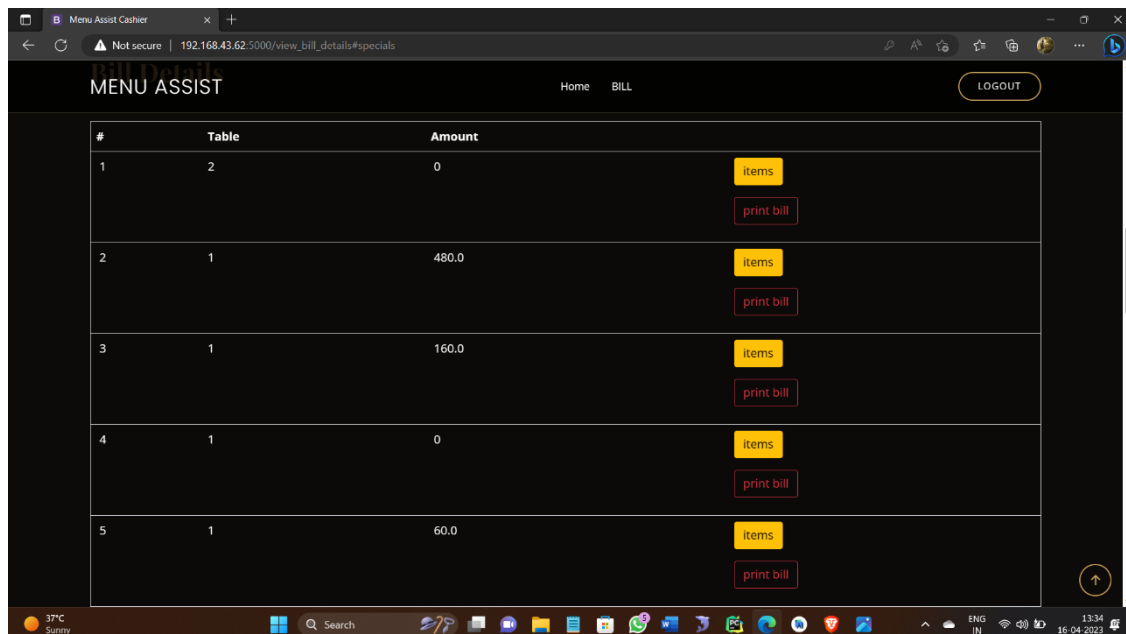
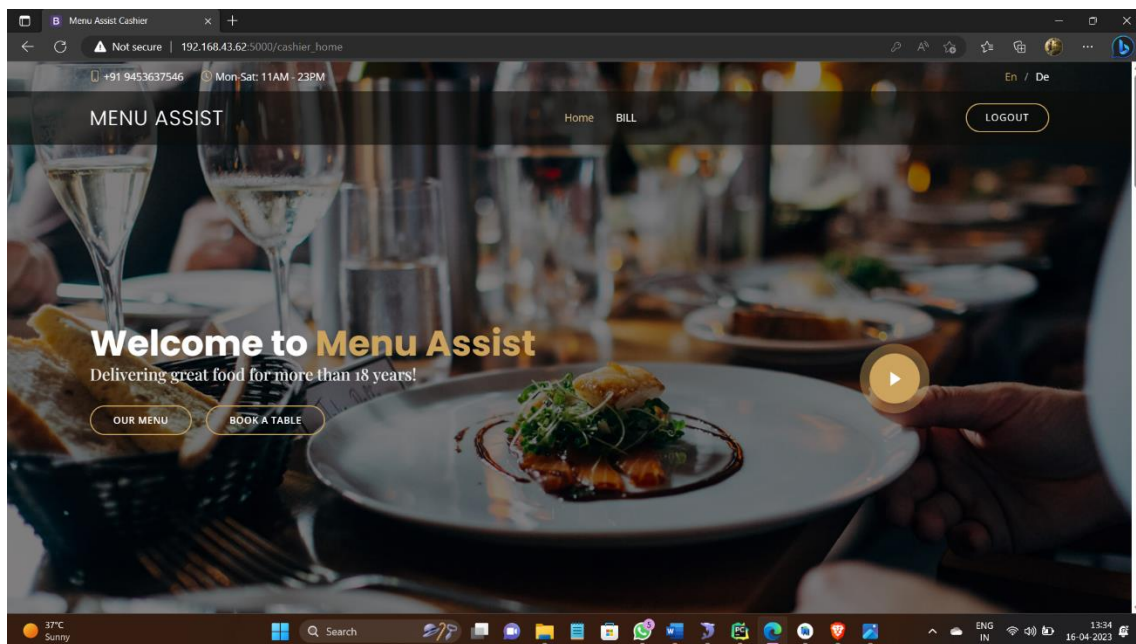




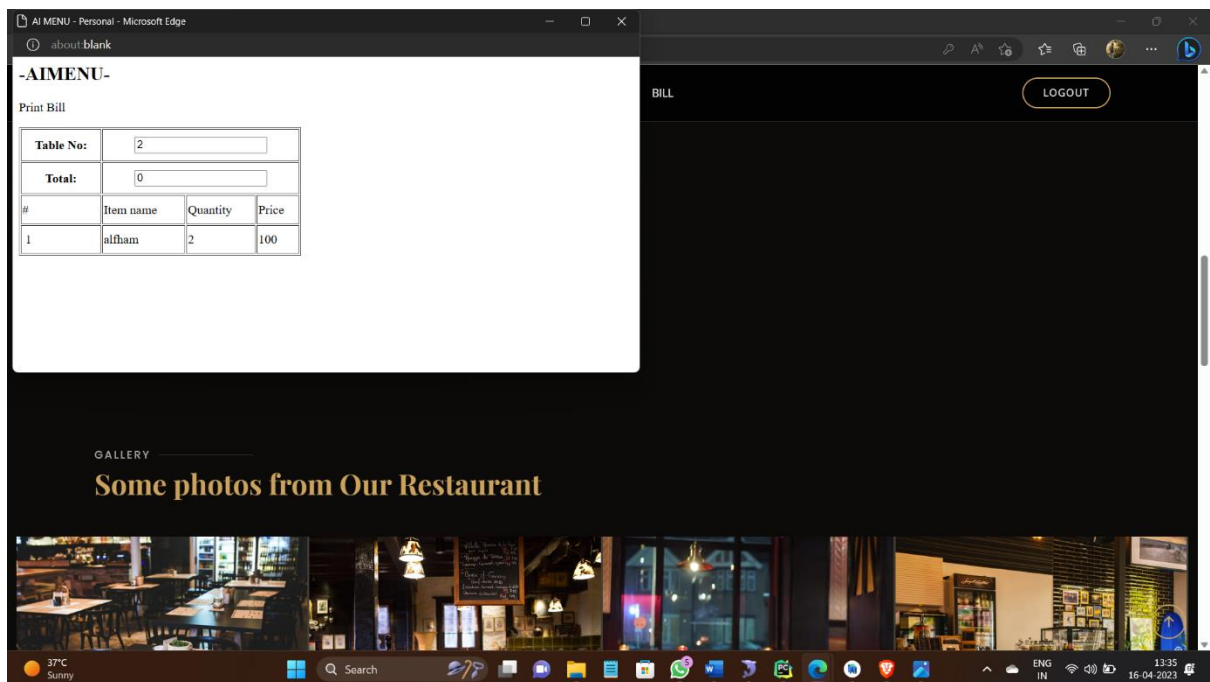
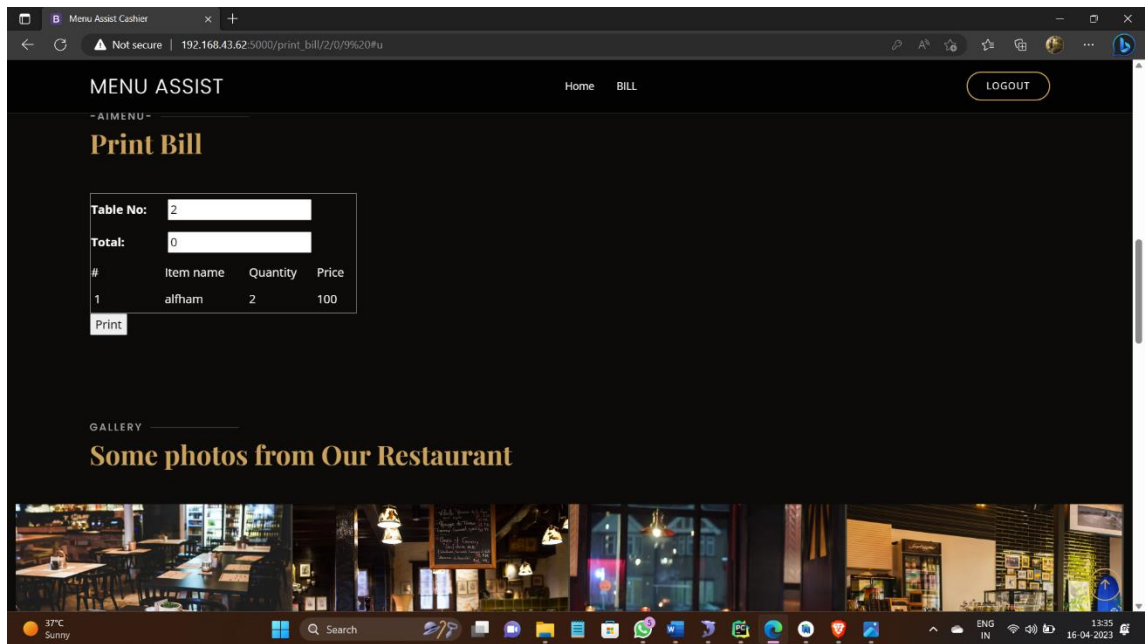
#### 10.1.4 ServiceStaff page:



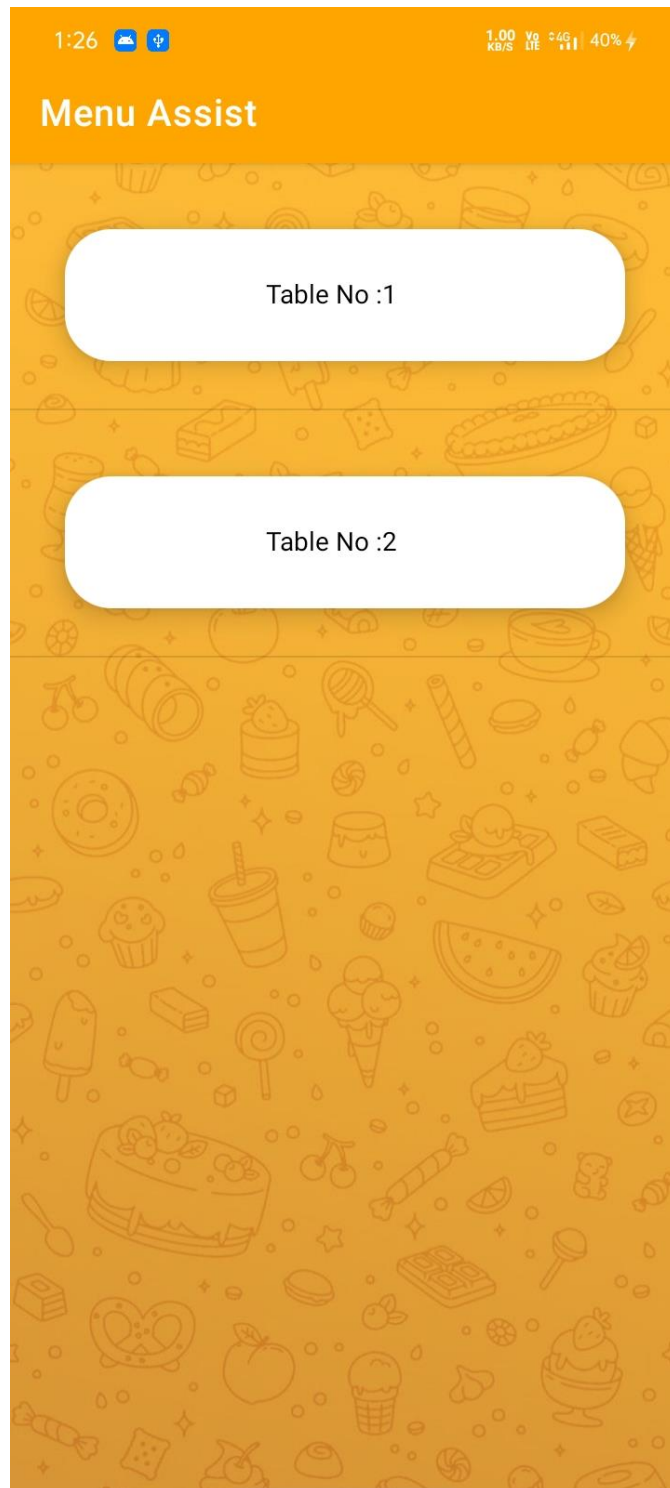
### 10.1.5 cashier page

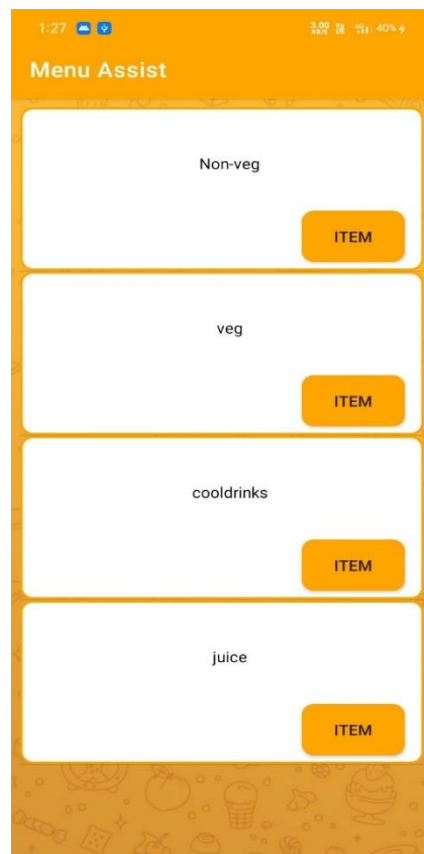


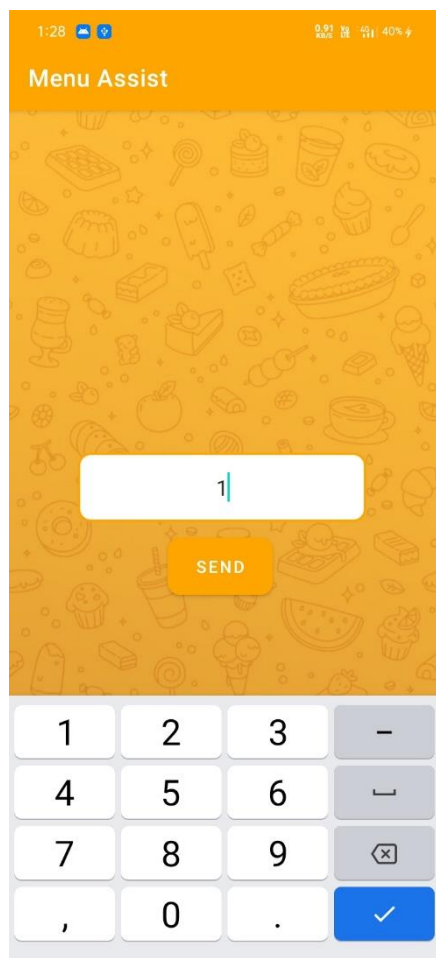
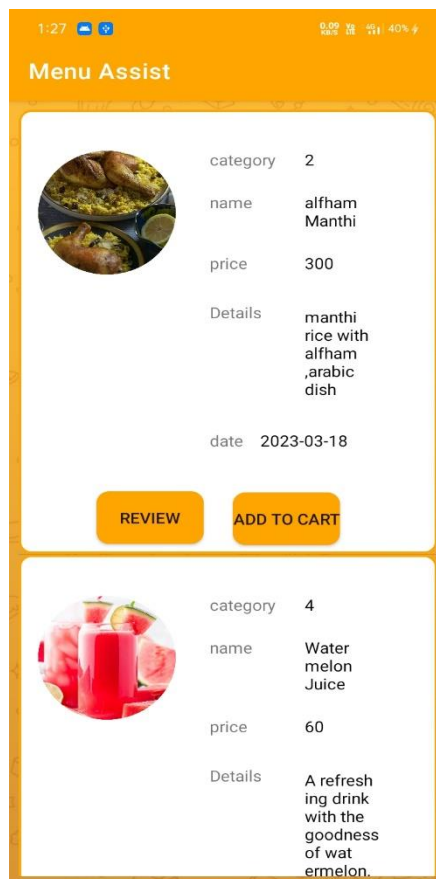




## 10.2 Android









1:28

8:57 信 告 40% 9

Menu Assist

Date	2023-04-16
Item name	Watermelon Juice
Item details	A refreshing drink with the goodness of watermelon. No preservation added.
Item Price	60
Amount	60
Quantity	1
Type	special_item

REVIEW

PLACE ORDER

9:30

信 告 72% 9

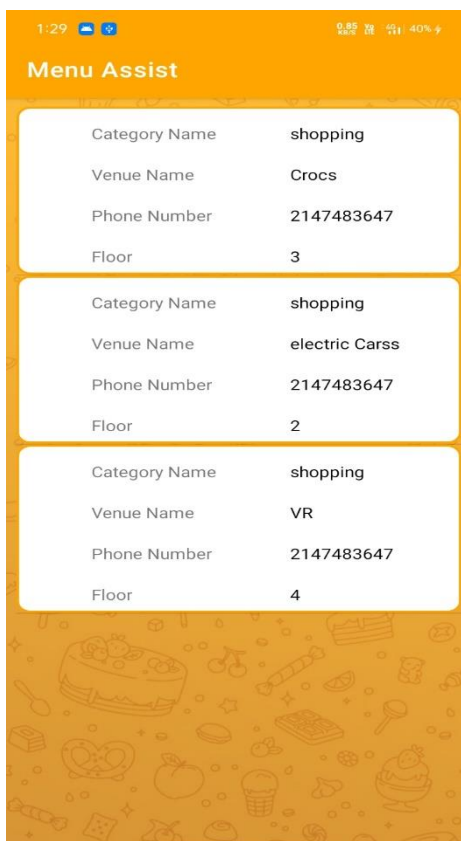
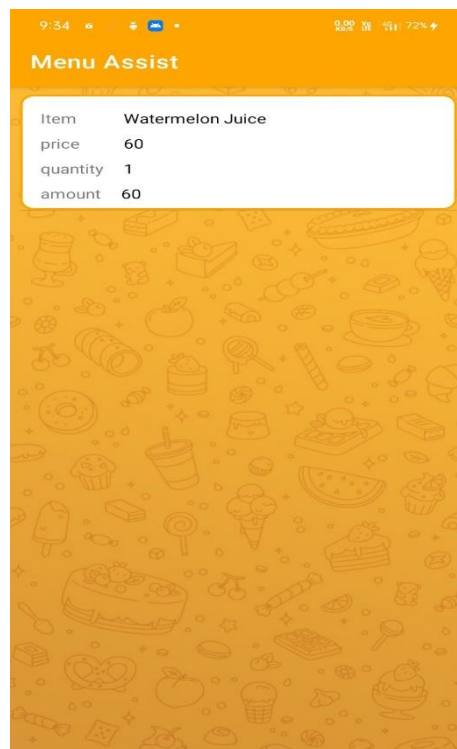
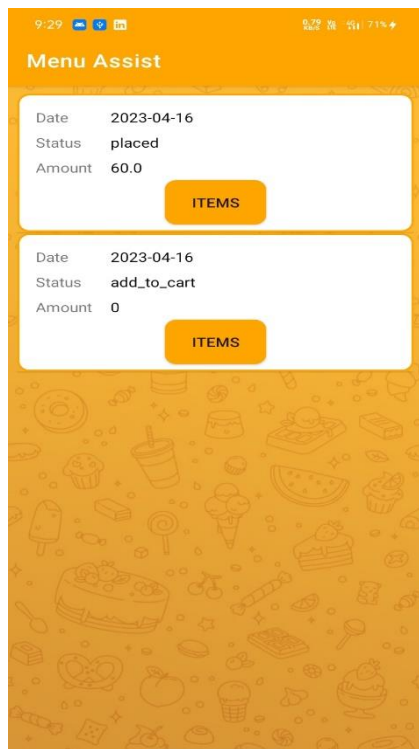
Menu Assist

Bank Name

Acc No

IFSC

PAY NOW



**A PROJECT REPORT ON**  
**Mizora Online Shopping & TradingApp**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ALEN ABRAHAM**

**REG.NO: DB20BCAR05**

**BIXON BENOY**

**REG.NO: DB20BCAR08**

**ASWIN SANDEEP**

**REG.NO: DB20BCAR24**



**DON BOSCO ARTS & SCIENCE COLLEGE ANGADIKADAVU,**  
**KANNUR, 670706**

**2023**

**A PROJECT REPORT ON**  
**Mizora Online Shopping & TradingApp**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ALEN ABRAHAM**

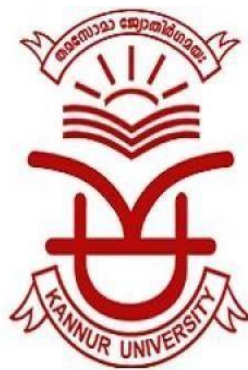
**REG.NO: DB20BCAR05**

**BIXON BENOY**

**REG.NO: DB20BCAR08**

**ASWIN SANDEEP**

**REG.NO: DB20BCAR24**



**DON BOSCO ARTS & SCIENCE COLLEGE ANGADIKADAVU,**  
**KANNUR, 670706**

**2023**

**DON BOSCO ARTS & SCIENCE COLLEGEANGADIKADAVU**  
**IRITTY, KANNUR**



**CERTIFICATE**

*Certified that this report titled\_Mizora Online Shopping & Trading App is a bonafide record of the project work done by ALEN ABRAHAM (REG.NO: DB20BCAR05), BIXON BENOY (REG.NO: DB20BCAR08) , ASWIN SANDEEP (REG.NO: DB20BCAR24) under the supervision and guidance ,towards partial fulfilment of the requirement for award of the degree of bachelor of computer application (BCA) of the Kannur university.*

Project Guide: SRUTHI N

Head of the Department

Angadikadavu

External Examiner

Date:

- 1.
- 2.

## **Declaration**

We ALEN ABRAHAM , BIXON BENOY , ASWIN SANDEEP sixth semester BCA students of Don Bosco Arts & Science College, Angadikadavu, under Kannur University do hereby declare that the project entitled “**MIZORA ONLINE SHOPPING AND TRADING APP** ” is the original work carried out by us in the sixth semester under the supervision of Ms. Sruthi N, Lecture of the Department of BCA, Don Bosco Arts & Science College, Angadikadavu, in partial fulfilment of the requirement for the award of the degree Bachelor of Computer Application, Kannur University.

Angadikadavu

ALEN ABRAHAM

BIXON BENOY

Date:

ASWIN SANDEEP

## **ACKNOWLEDGEMENT**

First of all we thank the lord almighty for his immense grace and blessings showered on me at every stages of this work. We are greatly indebted to our Principal Fr. Dr. Francis Karackat SDB, Don Bosco Arts & Science College, Angadikadavu for providing the opportunity to take up this project as part of our curriculum.

We deeply indebted to our project guide Mrs. Sruthi N, lectures of department of BCA, for her assistance and valuable suggestions as guide. She made this project a reality.

We express our sincere thanks to Mrs. Sindu PM, Mr. Hebin Layola, Mrs. Fincy Cyriac and Mrs. Vineetha Mathew, lecturers of department of BCA, for their valuable suggestions during the course of this project. Their critical suggestions helped me to improve the project work.

Acknowledging the efforts of everyone, their chivelorous help in the course of the project preparation and their willingness to collaborate with the work, their magnanimity through lucid technical details lead to the successful completion of my project.

We would like to express our sincere thanks to all our friends, colleagues, parents and all those who have directly or indirectly assisted during this work.

## CONTENTS

<b>chapters</b>	<b>contents</b>	<b>Page No</b>
1	INTRODUCTION	1
2	SYSTEM ANALYSIS	7
2.1	Existing System	8
2.1.1	Drawbacks Of Existing System	8
2.2	Proposed System	8
2.2.1	Advantage Of Proposed System	9
2.3	Feasibility Study	9
2.3.1	Economic Feasibility	10
2.3.2	Technical Feasibility	10
2.3.3	Behavioural Feasibility	10
2.4	System Specification	11
2.4.1	Software Specification	11
2.4.2	Hardware Specification	11
2.5	Identification Of Actors	12
2.6	Identification Of Use Cases	12
2.6.1	Use Cases For The Actor Administrator	13
2.6.2	Use Cases For The Actor User	14
2.6.3	Use Cases For The Actor Product Manager	15
2.6.4	Use Case Diagram	16



3	SYSTEM DESIGN	19
3.1	Introduction	20
3.2	JSON Document Database	20
3.3	Mizora Online Shopping And Trading App	22
3.4	Data Flow Diagram	24
3.5	ER Diagram	35
4	CODING	37
4.1	Input Interface	38
4.2	Output Interface	38
4.3	Software Description	38
4.3.1	HTML	38
4.3.2	DART	39
4.3.3	JavaScript	42
4.3.4	JSON	42
4.3.5	Node.js	43
4.3.6	MongoDB	45
5	CODING PAGES	47
5.1	Main Page	48
5.2	Router page	50
5.3	Account Screen page	54
5.4	Address Screen page	57
5.5	Admin.js	65

5.6	Admin.js Middleware	70
6	SYSTEM TESTING	71
6.1	Testing And Evaluation	72
6.2	Testing Strategies	73
6.3	Testing Techniques	74
6.3.1	White Box Testing	74
6.3.2	Black Box Testing	74
6.3.3	Unit Testing	75
6.3.4	Integration Testing	75
6.3.5	Acceptance Testing	76
6.3.6	Output Testing	76
7	SYSTEM IMPLEMENTATION AND DEPLOYMENT	77
8	CONCLUSION	79
9	REFERANCE	81
10	APPENDIX	82



## **1.INTRODUCTION**

## **1.1 Project Overview**

The Mizora Online Shopping & Trading App will primarily provide a platform to purchase, sell, distribution of items, product or service through the internet and on some other network. It will provide an option to a customer for the comparison of product with another seller, while a shop is available only at day time the e-commerce is available 24 hours of a day and seven days of the week. The e-commerce will be a huge marketplace as most of the business are going to implement based on the internet. This system will provide the detailed description of the products to users so that they can compare to the different product and will by the one which is more suitable to them.

## **NEED FOR THE SYSTEM**

The current E-commerce systems have various flaws in it as because of which most of the people don't use the system even though it has several advantages over traditional stores. The biggest problem is that it takes at least a day to deliver a product to the customer. While some other issues are a duplication of the product means the product is shown on the web some time differs with the original product due to which next time that customer go to buy the product through the traditional type of store.

## **OBJECTIVES AND SCOPE**

The E-Market platform has many advantages compare to traditional store as customer will get a correct market price of the product, can compare the cost of a product with other similar shopping applications ,if customer dislikes the product he/she can return it. We can use the current and advanced technologies and tactics to overcome the problems with the existing system.

It is possible to localize Mizora (proposed app) by providing a platform for small scale shops to display their geographical products and be as part of the network. Also with the help of these community we can deliver products (groceries, daily use products etc..) with in a day. With the application of machine learning the proposed system (Mizora) will develop a model which detect fake reviews posted by individuals and give a red notification about the reviews to all the users. A personalized search engine which is helpful for both the seller and user.

## **MODEL:**

The Agile model is a type of incremental model where software is developed in a rapid incremental cycle. The most notable feature of the Agile model is the establishment of the project scope, requirements, number, and duration of iteration at the beginning of the development process.

### **The Agile Model – Design:**

The meaning of Agile is swift or versatile. "**Agile process model**" refers to a software development approach based on iterative development. Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning. The project scope and requirements are laid down at the beginning of the development process. Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance.

Each iteration is considered as a short time "frame" in the Agile process model, which typically lasts from one to four weeks. The division of the entire project into smaller parts helps to minimize the project risk and to reduce the overall project delivery time requirements.

Each iteration involves a team working through a full software development life cycle including planning, requirements analysis, design, coding, and testing before a working product is demonstrated to the client.



**Fig. Agile Model**

### **Phases of Agile Model:**

Following are the phases in the Agile model are as follows:

1. Requirements gathering
2. Design the requirements
3. Construction/ iteration

### **Testing/ Quality assurance**

4. Deployment
5. Feedback

**1. Requirements gathering:** In this phase, you must define the requirements. You should explain business opportunities and plan the time and effort needed to build the project. Based on this information, you can evaluate technical and economic feasibility.

**2. Design the requirements:** When you have identified the project, work with stakeholders to define requirements. You can use the user flow diagram or the high-level UML diagram to show the work of new features and show how it will apply to your existing system.

**3. Construction/ iteration:** When the team defines the requirements, the work begins. Designers and developers start working on their project, which aims to deploy a working product. The product will undergo various stages of improvement, so it includes simple, minimal functionality.

**4. Testing:** In this phase, the Quality Assurance team examines the product's performance and looks for the bug.

**5. Deployment:** In this phase, the team issues a product for the user's work environment.

**6. Feedback:** After releasing the product, the last step is feedback. In this, the team receives feedback about the product and works through the feedback.

## **Agile Model – Application:**

### **Application of Agile model:**

Various software life cycle models in SDLC approach are available and used by all software development companies. Each software development application has different requirements based on internal and external factors. Agile model is the standard and upgraded SDLC MODEL. There have many requirements and useful situations where this agile model is highly suitable and handle the problem very smartly:

- Agile model is useful when system have rapidly new changes are needed to be implementation required. The freedom agile gives to change is very important. New changes can be implemented at very less cost because of the frequency of new increments that are produced.
- When to implement a new feature and the development team required losing only the work of a few days, or even only hours, to roll back and implement it.
- When both system developers and stakeholders alike to interact with the under development project.
- Applicable when developer required more freedom of time and options than if the software was developed in a more rigid sequential way.
- Provide ability to the authorized development team member about to leave important decisions until more or better data or even entire hosting programs are available; meaning the project can continue to move forward without fear of reaching a sudden standstill.



**The advantages of the agile model are as follows:**

1. Customer satisfaction is rapid, continuous development and delivery of useful software.
2. Customer, Developer, and Product Owner interact regularly to emphasize rather than processes and tools.
3. Product is developed fast and frequently delivered (weeks rather than months.)
4. A face-to-face conversation is the best form of communication.
5. It continuously gave attention to technical excellence and good design.
6. Daily and close cooperation between business people and developers.
7. Regular adaptation to changing circumstances.
8. Even late changes in requirements are welcomed.

**The disadvantages of agile model are as follows:**

1. It is not useful for small development projects.
2. There is a lack of intensity on necessary designing and documentation.
3. It requires an expert project member to take crucial decisions in the meeting.
4. Cost of Agile development methodology is slightly more as compared to other development methodology.
5. The project can quickly go out off track if the project manager is not clear about requirements and what outcome he/she wants.

## **2.SYSTEM ANALYSIS**

## **Introduction**

System analysis is the process of collecting and interpreting facts, understanding problems and using the information to suggest improvement on the system. This will help to understand the existing system and determine how computers make its operation more effective. The aim of this analysis is to collect detailed information on the system and the feasibility study of the proposed system.

### **2.1 EXISTING SYSTEM AND DRAWBACKS**

The existing systems does its prime job of digital commerce but it has some major disadvantages. The problems include lack of friendly user interface, security issues, clutters etc. It takes at least a day to deliver a product to the customer. These applications also led the local market to major downfalls dropping the business of small scale industries. Majority of customers buy products after checking the reviews. Even so, there is no way to detect fake reviews about the products which will drastically effect the business.

Existing system lacks the ability to use the capabilities of machine learning and AI to implement a immersive customer experience which include advanced search, price comparisons image recognition but most of the co- operates fear to implement these technique to their system as it take long procedures to implement it.

The existing systems doesn't have an offline mode and several other features that help the users to find out products easily which is missing from co-operative shopping apps

### **2.2 PROPOSED SYSTEM**

The Mizora platform has many advantages compare to traditional store as customer will get a correct market price of the product, can compare the cost of a product with other similar shopping applications if customer dislikes the product he/she can return it.

We can use the current and advanced technologies and tactics to overcome the problems with the existing system.

It is possible to localize Mizora (proposed app) by providing a platform for small scale shops to display their geographical products and be an part of the network. Also with the help of these community we can deliver products(groceries daily use products etc) with in a day.

With the application of machine learning the proposed system (Mizora) will develop a model which detect fake reviews posted by individuals and give a red notification about the reviews to all the users.

The proposed system integrates an image detector which help the customers to find out the exact product they needed, A personalized search engine which is helpful for both the seller and user. The proposed system implements features like image tagging, auto generated product description, customization, applications of VR

### **2.2.1 Advantages of Proposed System**

- Localization provides improved economy to society
- User friendly design with well-organized tabs
- Proper product evaluation and ratings
- Safe and assured data handling

### **2.3 Feasibility Study**

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioural feasibilities.

The proposed system is theoretically investigated to check the feasibility and found that they are more reliable and efficient in the cases given below. There are three aspects in the feasibility study portion of the preliminary investigation.

✓ Economic feasibility

✓ Technical feasibility

✓ Behavioural feasibility

The proposed system must be evaluated from a technical point of view first, and if technical feasible their impact on the organization must be assessed. If compatible, the operational system can be devised. Then they must be tested for economic feasibility.

### **2.3.1 Economic Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors which affect the development of a new system is the cost it would require. Since the system developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

### **2.3.2 Technical Feasibility**

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of accuracy, consistency, proper admin, backend functions. Having identified an outline system, the investigation must go on suggest the type of equipment, required method developing the system, of running the system once it has been designed. The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Though the technology become obsolete after some period of time, due to the fact that newer version of some software supports older versions, the system still be used. So there are only minimal constraints involved with this project. The system has been developed using dart programming language and flutter, along with Node JS as the backend with software MongoDB server, thus we could conclude that the project is technically feasible for development.

### **2.3.3 Behavioural Feasibility**

People are inherently resistant to change and devices have been known to facilitate change. The System is designed in user friendly manner and we no need to provide any special training for the persons using this software.

The operating system used is Android 11, which is also user friendly. Since the application can be easily accessed which is quite familiar to the intended users, it does not have any operational barriers. So no need to provide any special training for using this application software and hence it is behaviourally feasible.

## **2.4 System Specifications**

System Specification deals with the technical aspects the project has to meet in minimum to work successfully. This also includes the different aspects the software requirement is determined from. The technical details typically include:

- Software Specification
- Hardware Specification

### **2.4.1 Software Specifications**

The software required for the application depends on the following factors:

- ✓ The flexibility of the software
- ✓ Software contracts
- ✓ Limitation of the software

## **Software Requirement**

This specifies the minimum software requirements for implementing the system. This includes:

- Front End: - Dart, Flutter
- Back End: - Node JS, MongoDB
- Client side scripting: dart
- Server side scripting: java script
- Platform:-Visual Studio Code ,Flutter
- Operating System: Microsoft windows 10

### **2.4.2 Hardware Specifications**

The software required for the application depends on the following factors:

- ✓ Determining size and capacity requirements.
- ✓ Computer evaluation and measurement.

- ✓ Financial factors.
- ✓ Maintenance and support.

## **Hardware Requirement**

- Microprocessor: Any 64 bit processor.
- Clock speed: - 2.13GHz
- Ram: 1 GB and above
- Hard disk: 40 GB and above
- Keyboard:-standard keyboard
- Mouse: Standard mouse
- Connectivity: - LAN & Wi-Fi
- Camera: Standard Camera

## **2.5 Identification of Actors**

A use cases represents the functionality of an actor. It is defined as a set of actions performed by a system, which yields an observable result. An ellipse containing its name inside the ellipse or below it represents. it. It is placed inside the system boundary and connected to an actor with an association. This shoes how the use cases and the actor interact.

We can identify the actors through a list of questions. The answers to these questions bring out the actors of the system is.

- Admin
- User
- Product Manager

Here we need to specify the use cases of each actor.

## **2.6 Identification of use cases**

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yield an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

To find out the use cases, ask the following questions to each of the actors.

- ✓ Which functions does the actor require from the system? What does the actor need to do?
- ✓ Does the actor need to read, create, destroy, modify or store some kind of information in the system?
- ✓ Does the actor have to calculate something? And want to provide information for others?
- ✓ Could the actor's daily work be simplified or made more efficient by adding new functions to the system (typically functions which are currently not automated in the system)?

### **2.6.1 Use cases for the actor Administrator**

#### **Login:**

The first step involved is login. The admin can login to the website using username and password.

#### **Add/Delete Products:**

Admin can add or remove products.

#### **Analyse user Traffic:**

Admin can manage the users and app traffic

#### **Picking Images:**

Picking images for the products.

#### **Fetch/Display Products:**

Display the products for users.

#### **Change Order Status:**

Admin can change and update order status for the users

**View Orders:** Admin can view the orders.



**Payment Handling:**

Admin incorporates payment services.

**Send Notification:**

Admin can send notification.

**Change password:**

Admin can change password.

**2.6.2 Use cases for the actor User****Sign Up:**

The user can sign up to the application.

**Login/Logout:**

User can login using username and password and also log out.

**Search Product:**

User can search for various products.

**Add products to Cart:**

User can add product to the cart.

**Select Quantity:**

User can select the quantity of items.

**Rating Products:**

User can rate products.

**Payment:**

User can pay for the products via payment portal.

**Manage Account:** User can manage his/her account.

### **2.6.3 Use cases for the actor Product Manager**

#### **Display Sales Chart:**

Display the sales chart of the business status for the admin.

#### **State Persistence:**

Product manager can automatically login the identified users.

#### **Fetch Average Ratings:**

Product manager can calculate the average rating for products.

#### **Display Trending Products:**

Product manager can highlight trending products based on ratings.

#### **Admin/User Authentication:**

Product manager authenticates admin and users.

#### **Product price calculations:**

Calculate the prices of various products.

#### **Address Validation:**

Validation of the address.

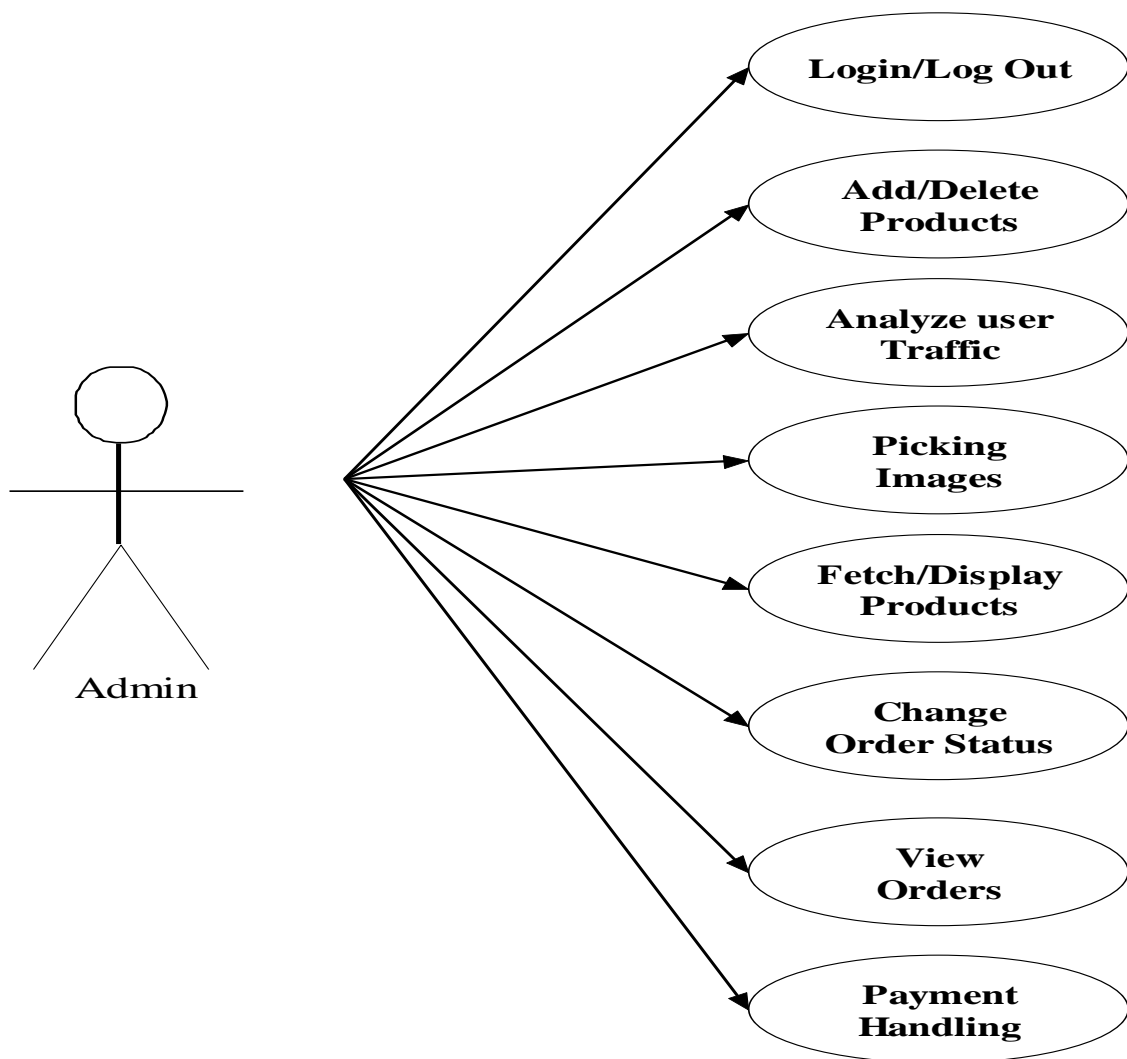
#### **Fetching Order Details:**

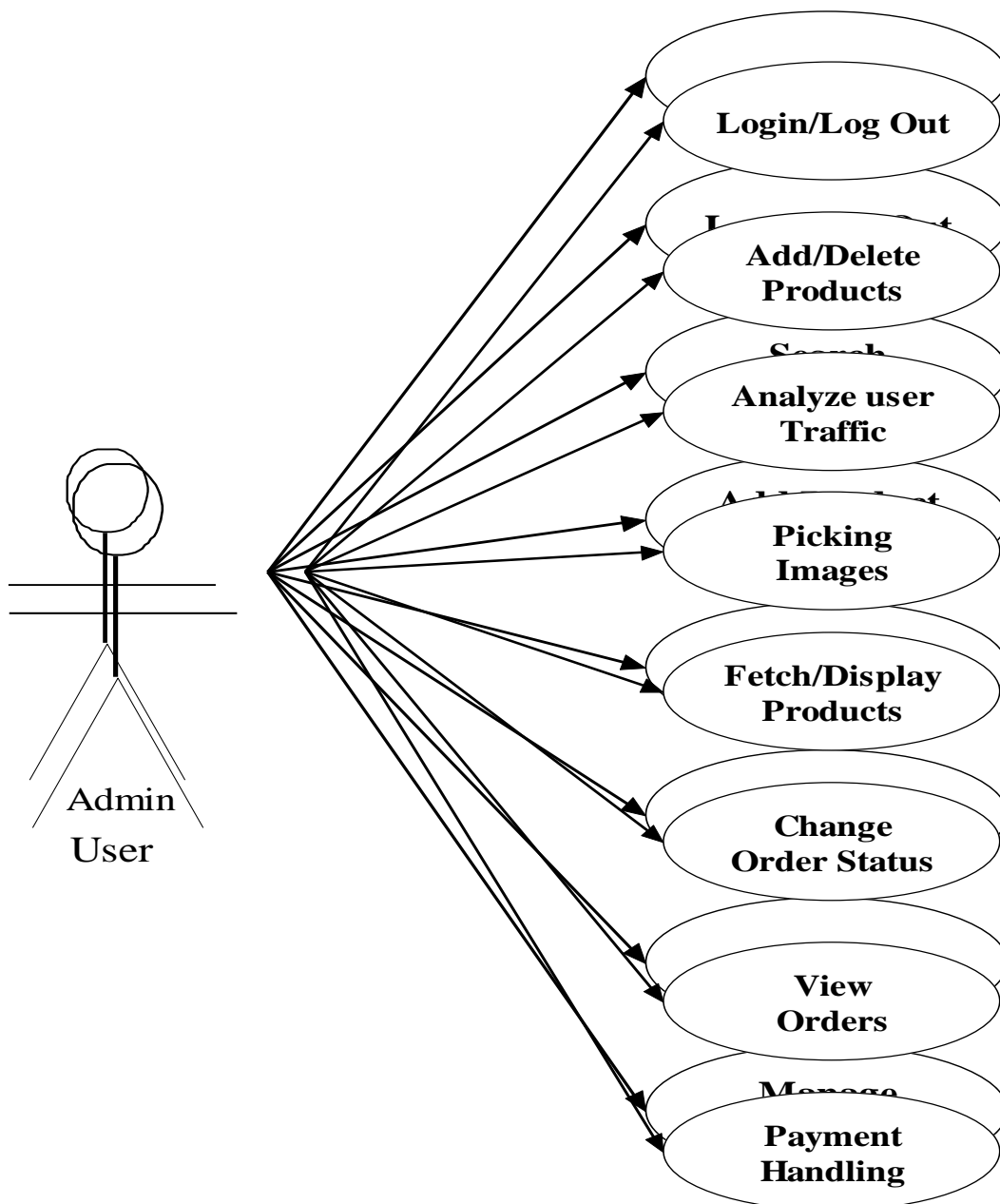
Product manager fetches the order details.

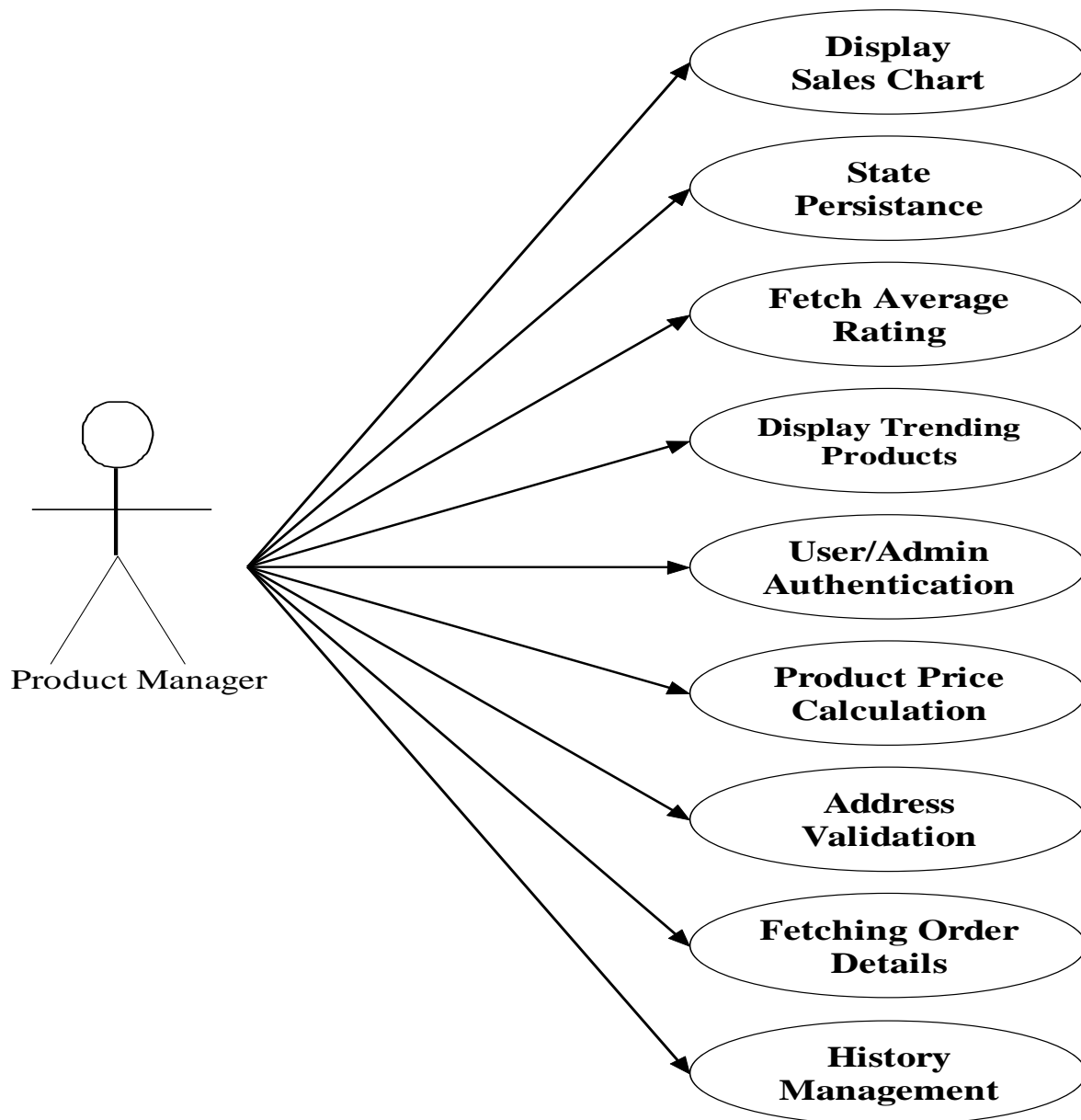
#### **History Management:**

Product manager handle the user history.

## 2.6.7 USE CASE DIAGRAM







## **2. SYSTEM DESIGN**

### **3.1 INTRODUCTION:**

System design provides an understanding of the procedural details, necessary implementing the system recommended in the feasibility study. Basically it is all about the creation of a new system. This is a critical phase since it decides the quality of the system and has a major impact on the testing and implementation phases.

**System design consists of three major steps.**

- Drawing of the expanded system data flow charts to identify all the processing functions required.
- The allocation of the equipment and the software to be used.
- The identification of the test requirements for the system.

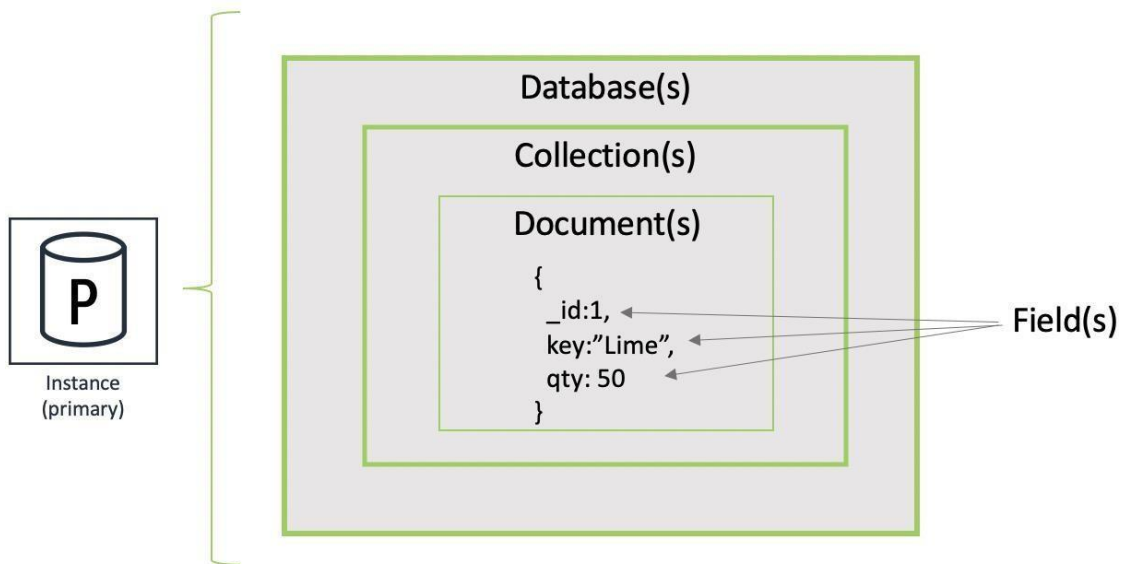
### **CHARACTERS OF DESIGN**

- A design should exhibit a hierarchical organization that makes intelligent use of control among components of the software.
- A design should be modular that is, the software should be logical.
- A design should contain distinct and separable representation of data and procedure.
- A design should lead to interface that reduce the complexity of the connections between modules and with the external environment.

### **3.2 JSON Document Database**

JSON document database is a type of non-relational database that is designed to store and query data as JSON documents, rather than normalizing data across multiple tables, each with a unique and fixed structure, as in a relational database. JSON document databases use the same document-model format that developers use in their application code, which make it much easier for them to store and query data.

The flexible, semi-structured, and hierarchical nature of JSON document databases allows them to evolve with applications' needs. JSON document databases provide powerful and intuitive APIs for flexible and agile development.



Database	Collection	Command	Field	Operator	
↓	↓	↓	↓	↓	
db.products	.find	( {	qty: { \$gt: 4 }	})	→ { "_id": 1, "key": "Lime", "qty": 50 }
		(query)			(document)



### 3.3 Mizora Online Shopping And Trading App

#### Add user

\_Id:

63afd43d5a45007e91e32bfe

Name: "bixon"

Email: "bixon@gmail.com"

Password: "\$2a\$08\$ubAt45DHnvgYlK40ru4o.Zk.kgBB1zaneP1pLy09gaie/Pmmb6g."

Address: "nosie,oisgg,kannur-75686"

Type: "user"

v:46

Cart: Array

## **Add Product**

\_Id:6418694c41fb1a436a850a75

Name: "Kudapura rotti"

Description: "Kori rotti is a spicy dish of Tulu Udupi-Mangalorean cuisine, a combin..."

Images:"https://res.cloudinary.com/djvmcncow/image/upload/v1677741034/PANTS/r7..."

Quantity: 20

Category: "Localization"

Price:30

Ratings: 4\*

### **3.4. Data Flow Diagram**

A graphical representation is used to describe and analyses the movement of data through a system manual or automated including the processes, Storing of data and delays in the system. Data flow diagrams are the central tool and the basis from which other components are developed.

The transformation of data, from input to output through process may be described logically and independently of the physical components associated with the system.

They are termed logical dataflow diagrams, showing the actual implementations and the movement of data between people, departments and

workstations. DFD is one of the most important modelling tools used in system design. DFD shows the flow of data through different process in the system.

#### **PURPOSE:**

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have reasonably completed model of the behavior of the system. It is important to avoid premature designs, wherein develop designs before analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

Throughout my project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In my opinion, "efficient design of the data flow and context flow diagram helps to design the system successfully without much major flaws within the scheduled time". This is the most complicated part in a project. In the designing process, my project took more than the activities in the software lifecycle. If we design a system efficiently with all the future enhancements the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as a way of expressing system requirements in graphical form. A data flow diagram also known as "bubble chare" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It functionality decomposes the requirement specification down to the lowest level.

Data Flow Diagram depicts the information flow, the transformation flow and the transformations that are applied as data move from input to output. Thus DFD describes what data flows rather than how they are processed.

Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is one of the most important tools used during system analysis. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

Data Flow Diagrams are made up of a number of symbols, which represents system components. Data flow modelling method uses four kinds of symbols, which are used to represent four kinds of system components.

These are

- Process
- Data stores
- Data flows
- External entity

#### **Process:**

Process shows the work of the system. Each process has one or more data inputs and produce one or more data outputs. Processes are represented by rounded rectangles in Data Flow Diagram. Each process has a unique name and number. This name and number appears inside the rectangle that represents the process in a Data Flow Diagram.

#### **Data Stores:**

A data stores is a repository of data. Processes can enter data, into a store or retrieve the data from the data store. Each data has a unique name.

#### **Data Flows:**

Data flows show the passage of data in the system and are represented by lines joining system components. An arrow indicates the direction of flow and the line is labelled by name of the dataflow.

### **External Entity:**

External entities are outside the system but they either supply input data into the system or use other systems output. They are entities on which the designer has control. They may be an organizations customer or other bodies with which the system interacts. External entities that supply data into the system are sometimes called source. External entities that use the system data are sometimes called sinks. These are represented by rectangles in the Data Flow Diagram.

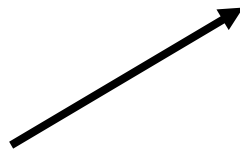
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Basic data flow diagram symbols are.....

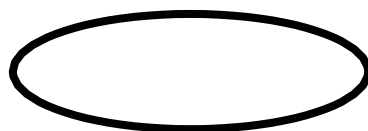
- A Square defines a source (originator) or destination of a system data:



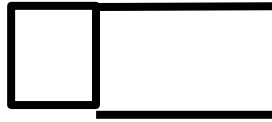
- An Arrow identifies data flow. It is a pipeline through which information flows:



- A Circle represents a process that transforms incoming data flow(s) into outgoing data flow(s):



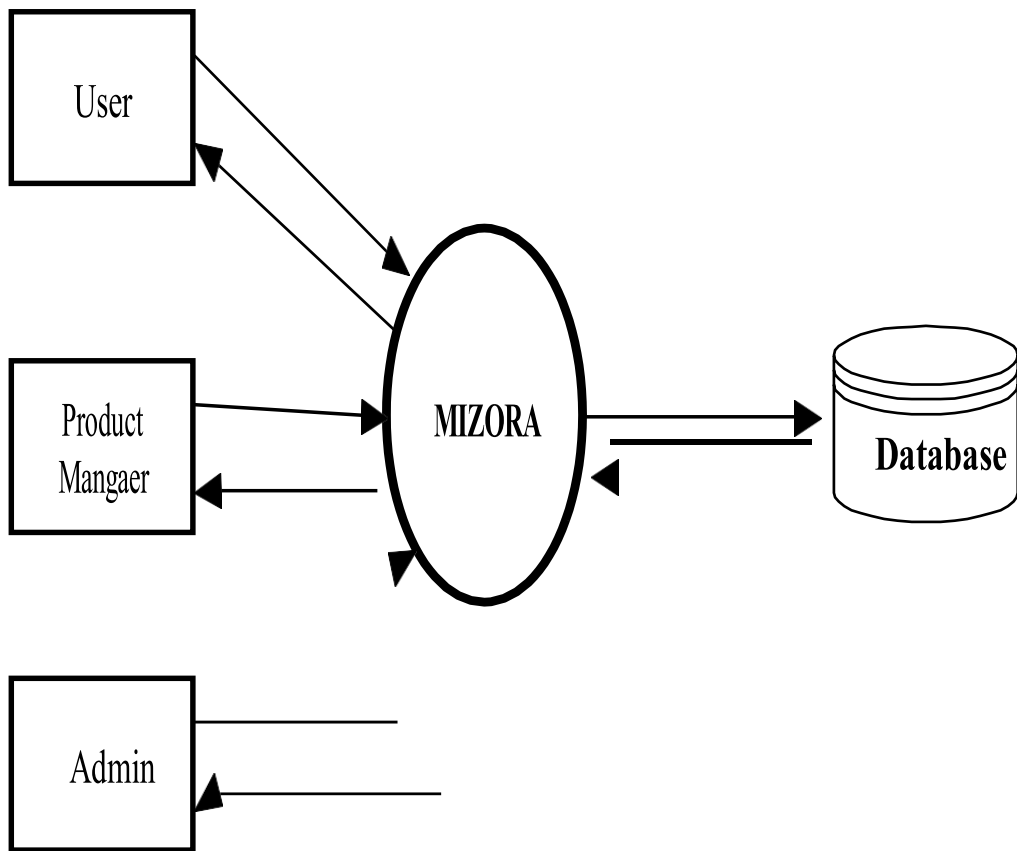
- An Open Rectangle is a data store:



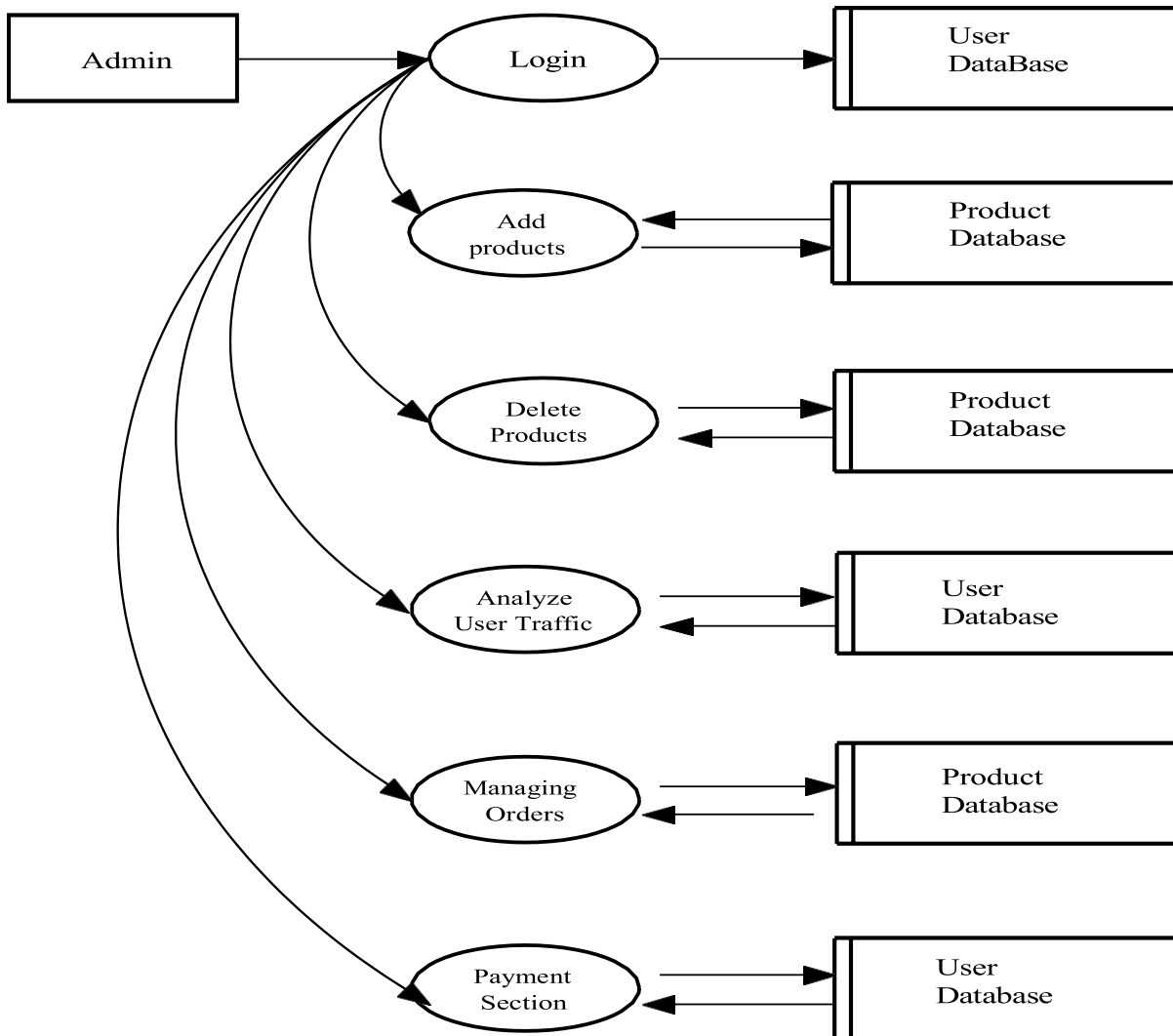
**Four steps are commonly used to construct a DFD:**

- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and left to right.
- When a process is exploded in to lower level details they are numbered.
- The names of data stores, sources and destinations are written in Capital letters.

### DFD Level-0

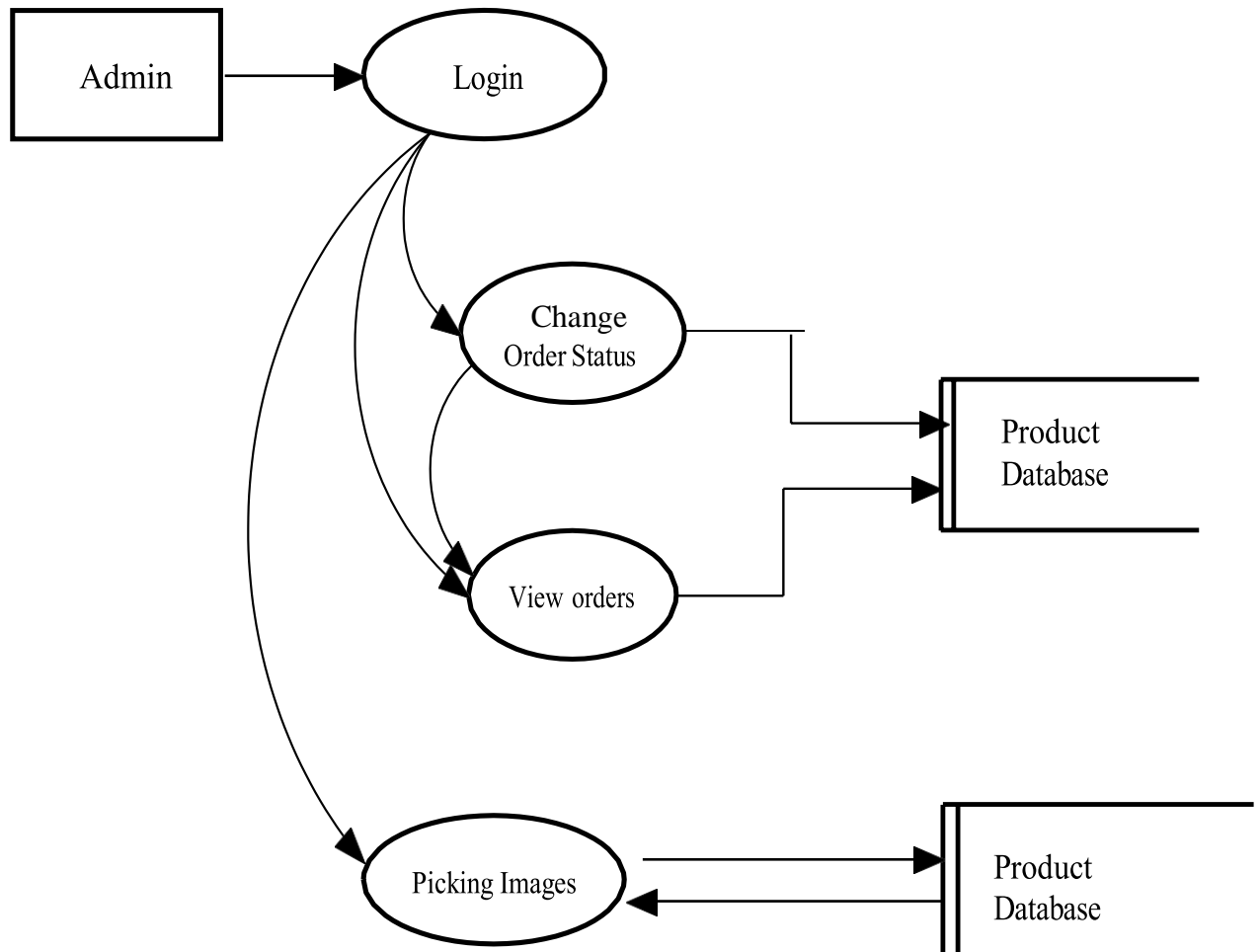


### DFD Level – 1 (Admin)

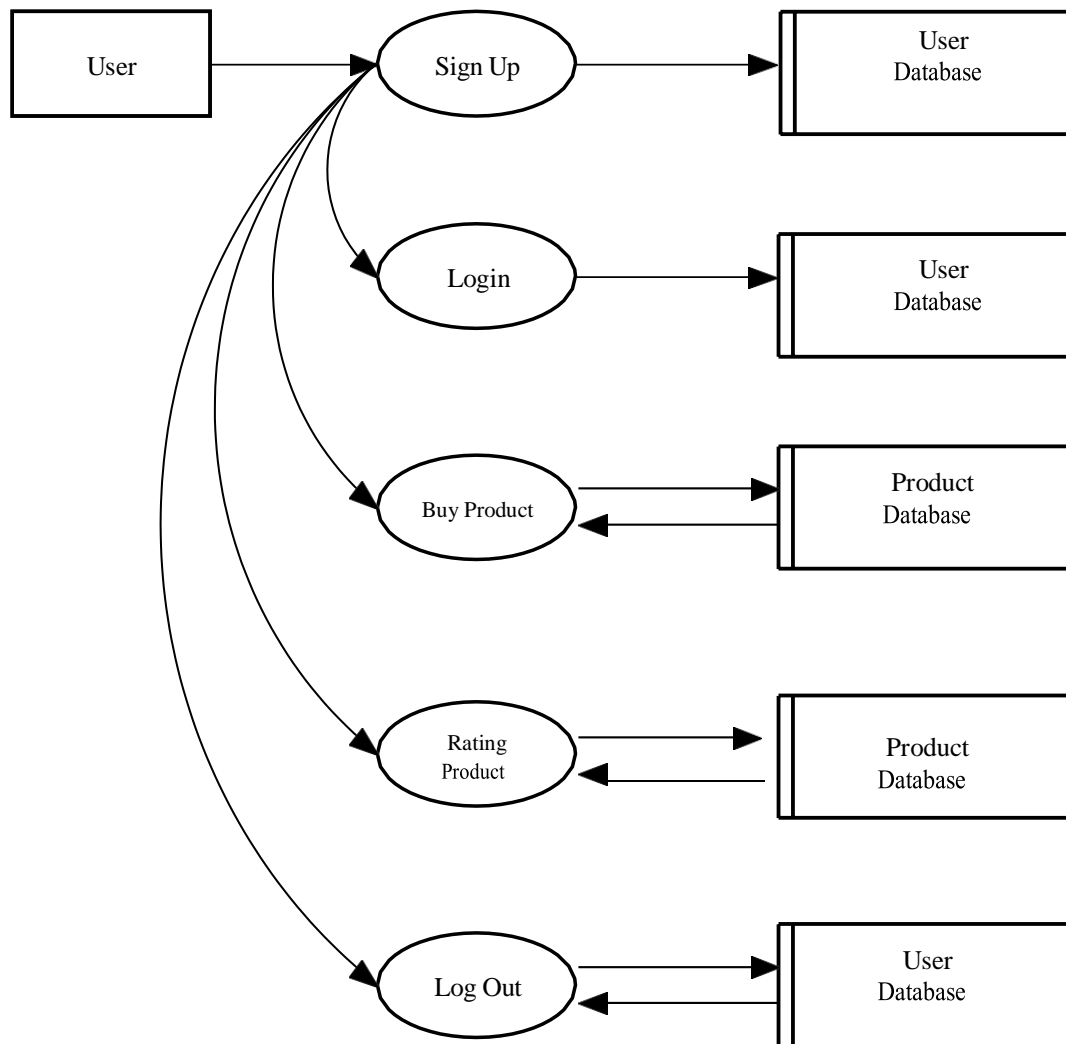




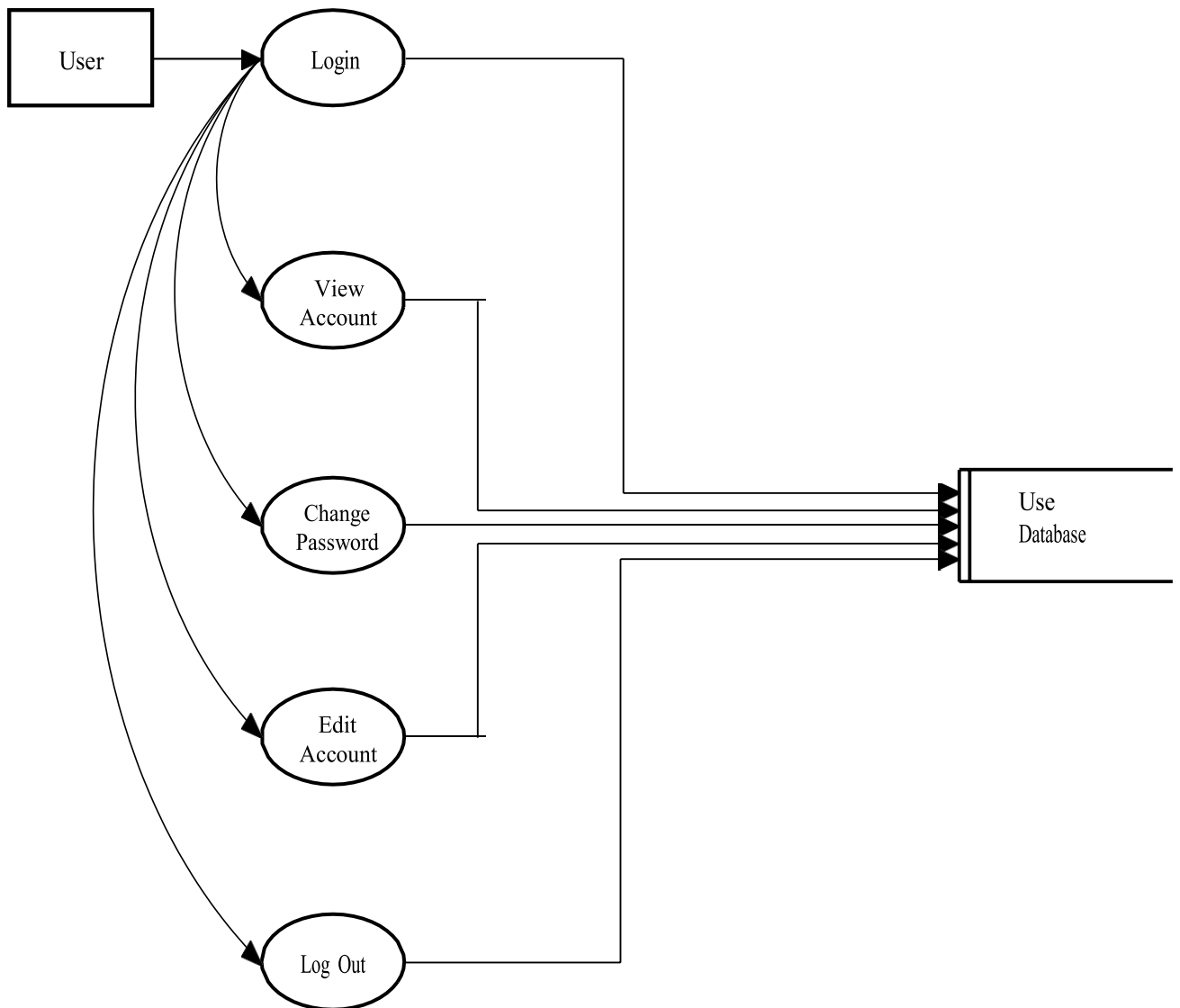
### DFD Level – 1.1 (Admin)



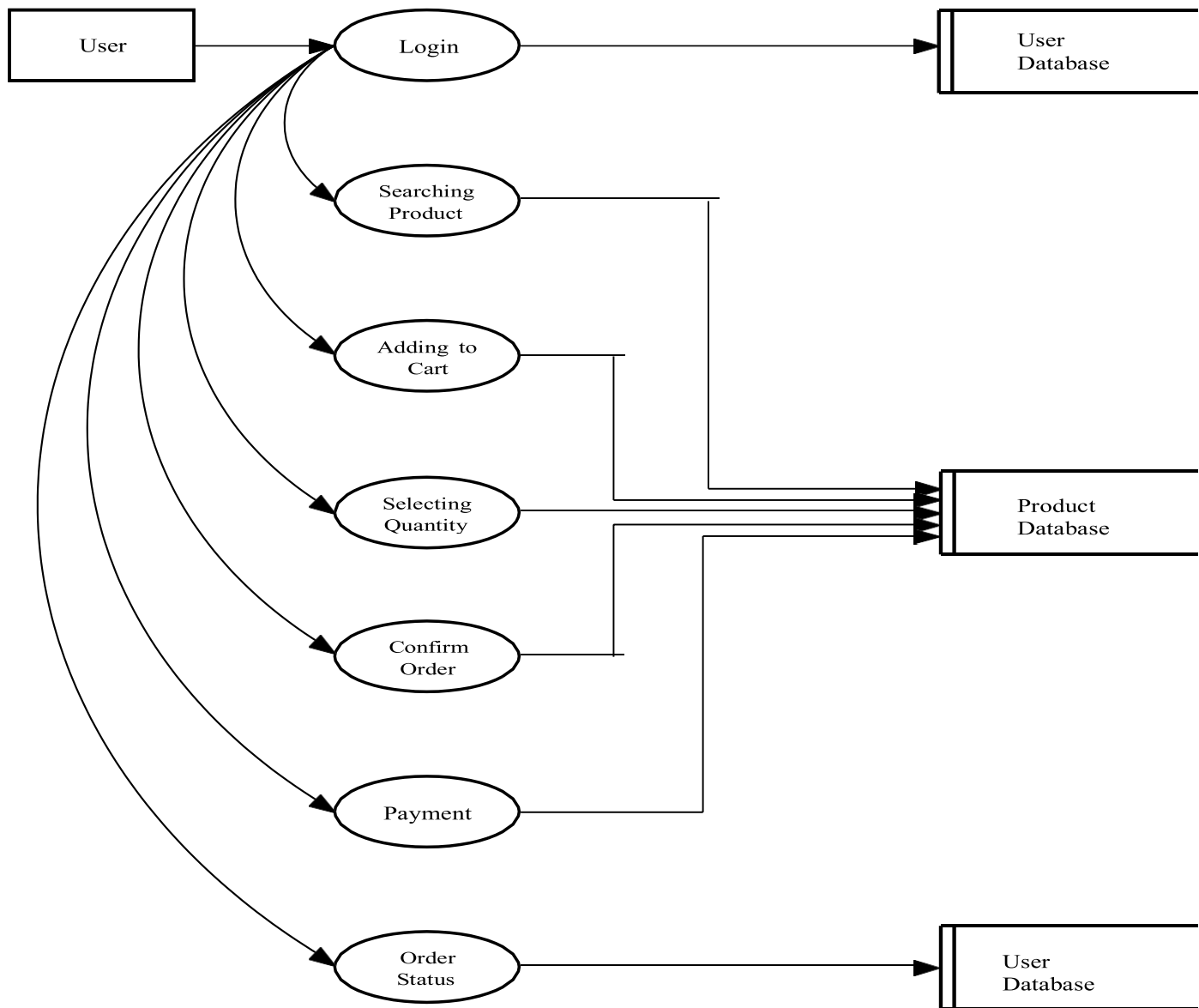
### DFD Level – 2 (User)



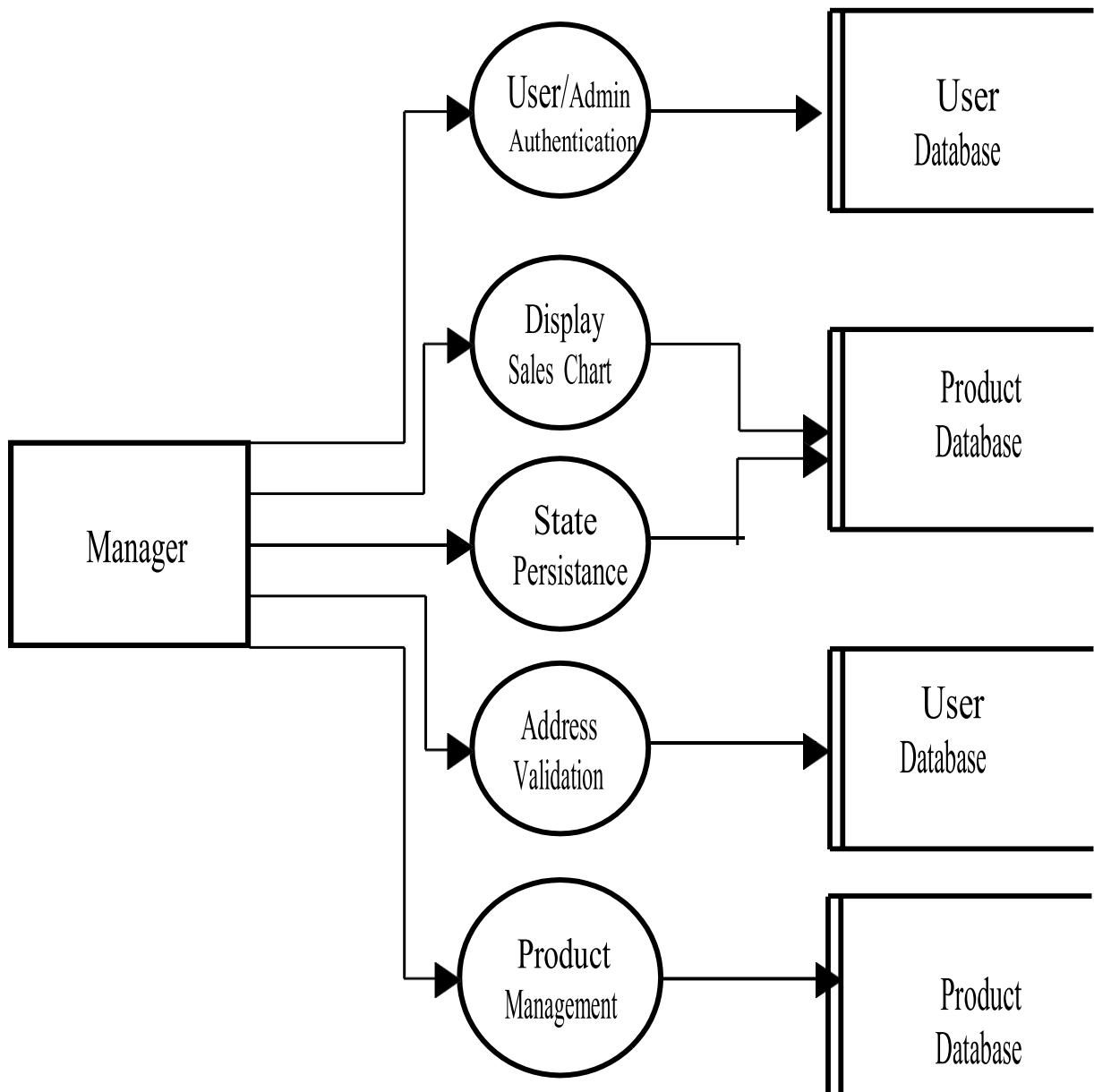
## **DFD Level – 2.1 (User)**



## DFD Level – 2.2 (User)



### DFD Level- 3 (User)



### 3.5 ER Diagram

An ER diagram is a diagram that helps to design databases in an efficient way. It is a data model for describing the data or information. It is a visual representation of data that describes how data is related to each other. The main components of ER models are entities, attributes and the relationships that can exist among them.

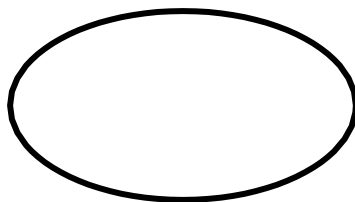
#### Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.



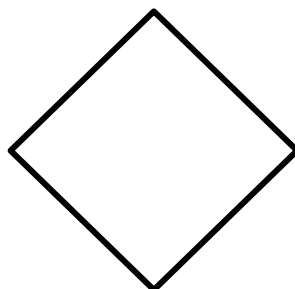
#### Attribute

Attributes are properties of entities. Attributes are represented by means of eclipses. Every eclipse represents one attribute and is directly connected to its entity (rectangle).

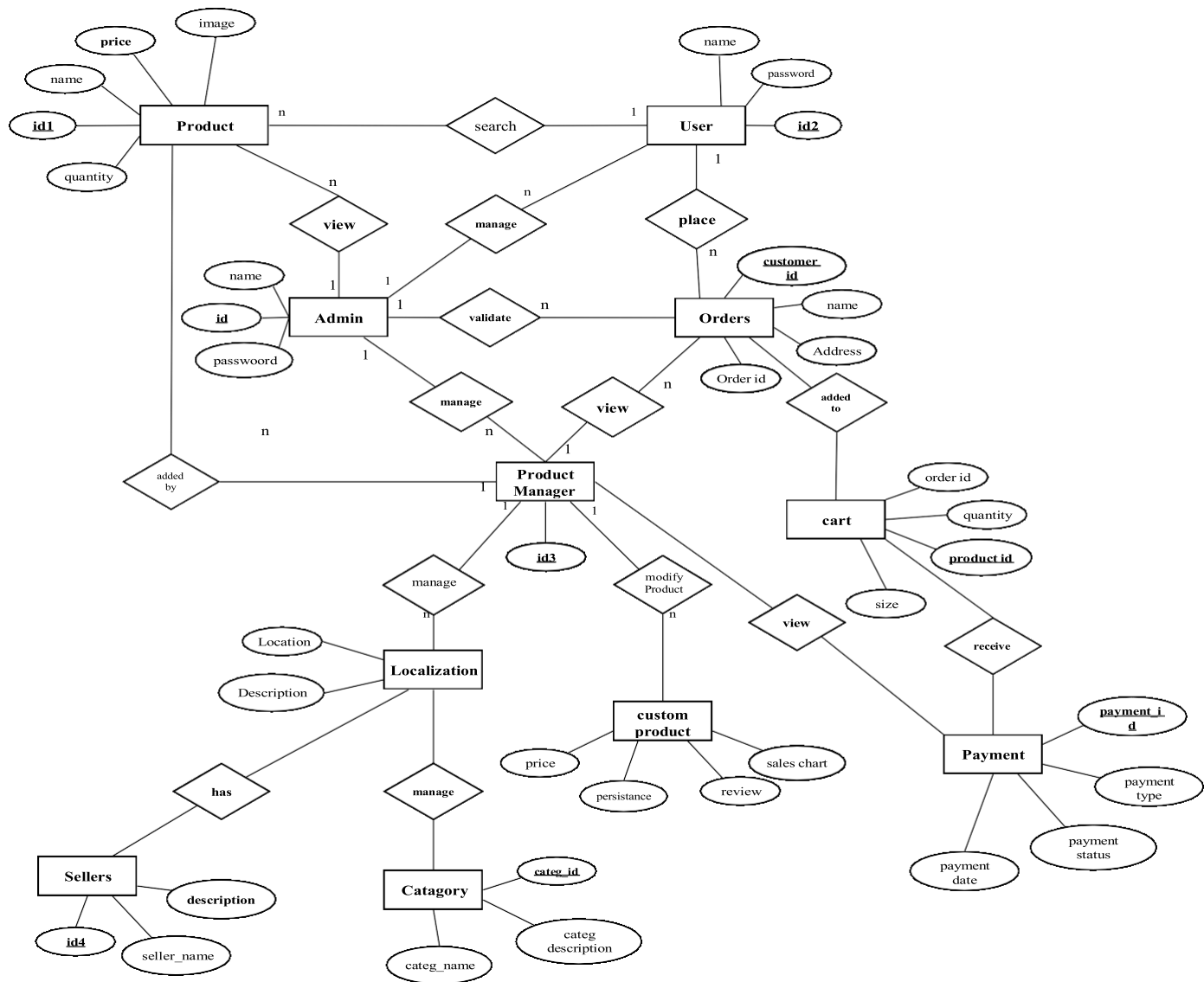


#### Relationship

Relationships are represented by diamond shaped box. Name of the relationship is written in the diamond box. All entities (rectangles), participating in relationship, are connected to it by a line.



## Architectural Design



## **4. CODING**



## **4.1 INPUT INTERFACE**

Input design is a part of overall system design, which requires very careful attention. If data going into the system is correct, then the processing and output will magnify these errors. Thus the designer has a number of clear objectives in the different stages of input design.

- To produce a cost effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that input is acceptable to and understood by the user.

## **4.2 OUTPUT INTERFACE**

At the beginning of the output design various types of outputs such as external, internal, operational and interactive and turn around are defined. Then the format, content, location, frequency, volume and sequence of the outputs are specified. The content of the output must be defined in detail. The system analysis has two specific objectives at this stage.

- To interpret and communicate the results of the computer part of a system to the users in a form, which they can understand, and which meets their requirements.
- To communicate the output design specifications to programmers in a way in which it is unambiguous, comprehensive and capable of being translated into a programming language.

## **4.3 SOFTWARE DESCRIPTION**

### **4.3.1 HTML**

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

HTML files are written in ASCII text, so the user can use any text editor to create his/her web page, though a browser of one sort or another is necessary to view the web page. HTML is case insensitive with its language commands. The characters within the document, however, are case sensitive. The language consists of various "tags" which are known as elements. These allow the browser to understand (and put into the desired/specified format) the layout, background, headings, titles, lists, text and/or graphics on the page. The elements are classified according to their function in the HTML document. There are head elements and body elements. The head elements identify properties of the entire document, while body elements actually mark text as content and show a change in the appearance in one way or another. Most elements have a beginning and an ending which encompass the text the user wishes to mark with the tag. All HTML documents must begin with the element and end with the element. Some of the other elements which may be used are tags to create lists-- both ordered lists as well as unordered lists. The user may also create larger or smaller, bolder, italicized, or underlined text. Attributes may be used along with the elements. These perform functions such as placement of text, indication of the source files of images, and identification of links to the document or part of the document.

#### **4.3.2 Dart**

Dart is an open-source, general-purpose, object-oriented programming language with C-style syntax developed by **Google in 2011**. The purpose of Dart programming is to create a frontend user interfaces for the web and mobile apps. It is under active development, compiled to native machine code for building mobile apps, inspired by other programming languages such as Java, JavaScript, C#, and is Strongly Typed. Since Dart is a compiled language so you cannot execute your code directly; instead, the compiler parses it and transfer it into machine code.

It supports most of the common concepts of programming languages like classes, interfaces, functions, unlike other programming languages. Dart language does not support arrays directly. It supports collection, which is used to replicate the data structure such as arrays, generics, and optional typing.

Despite being a relatively new programming language, Dart has gained immense popularity due to its unique features and user-friendly attributes.

## **Advantages of Dart**

### **1. Easy to learn language**

Dart is a fairly easy language to learn, and Google developers have put in a tremendous effort in the documentation part. With its Java-like syntax, developers with OOPS background can quickly plunge into programming if they know the basics. Dart also allows for easy editing as they can test small sections of code even if the complete application is not ready yet. Dart is fairly easy to grasp, modern, functional, flexible and competitive. The ecosystem is simple, understanding the terminologies, the proper tools and SDKs for the language is easy, and accessing the frameworks and libraries is easier. If a developer is familiar with any programming language, not just necessarily an OOP language, they can intuitively start using Dart.

### **2. Comes with good documentation**

Developers find that Dart is a good first programming language to learn because it has an excellent introduction and very good documentation. Getting started is also easy; just type the Dartpad url, and you can get started. More and more people have switched to Dart, thanks to its simple syntax, excellent community support, easy features that guide developers when they are in the training process.

### **3. High performance factor**

Applications run in Dart run faster than in other programming languages. And features like JIT compilation and AOT compilation add to the performance feature of Dart. JIT compilation or Just in Time compilation helps you to enable hot reloads, while AOT or Ahead of Time compilation helps with fast startup and better execution of the app.

### **4. Dart syntax is clean**

Dart looks almost similar to Java as it has clean syntax. So developers can easily pick up the code easily, but there is a chance they could get confused with many Dart language features.

## **5.Excellent tooling support**

The programming language has incredible tools to support app development. While looking at the advantages, you must be aware of the drawbacks as well, so as to help make a wiser decision.

## **6.Can compile to self-contained snapshots**

This feature is possible with other languages, but it is fast and simple with Dart. The Dart scripts can compile to self-contained snapshots on its own, i.e without requiring any other programs or libraries.

## **7.Can write the first program without installation or configuration**

Dart comes with DartPad, a very simple interface, eliminating the need of installation or configuration. Just write the code, and click on Run command to execute the code. There is support for libraries, but it's restricted to basic level.

## **8.A good support for the programmer**

Programmers can opt to treat Dart as an ordinary, dynamically typed language, especially if they do not want to deal with type systems at all. So Dart is an optionally typed language. Developers can also benefit from the extra documentation that comes along with the type annotations in the code. Dart alerts the programmers about possible type inconsistencies and oversights and not errors. These alerts are calibrated to support the developers.

## **9.More type-safe than Javascript**

If you compare Dart with Javascript, the former has a few advantages in certain aspects. For example, Javascript is not a type-safe language. It is only during the run-time, will you see the programming errors. On the other hand, Dart supports both strong and loose prototyping, where you can see the programming errors during compilation. So it is more type-safe than JS.

## **10.Dart is portable**

There is no need for any specific hardware configurations or architecture for running Dart, as it functions on any operating system and in all web browsers.

### 4.3.3 JAVASCRIPT

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript.

The General-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

Advantages of JavaScript:

- Less server interaction – you can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.
- Immediate feedback to the visitors – They don't have to wait for a page reload to see if they have forgotten to enter something.
- Increased interactivity – you can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.
- Richer interfaces – you can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

### 4.3.4 JSON

JSON is an acronym for JavaScript Object Notation, is an open standard format, which is lightweight and text-based, designed explicitly for human-readable data interchange. It is a language-independent data format. It supports almost every kind of language, framework, and library. In the early 2000s, JSON was initially specified by Douglas Crockford. In 2013, JSON was standardized as ECMA-404, and RFC 8259 was published in 2017.

JSON is an open standard for exchanging data on the web. It supports data structures like objects and arrays. So, it is easy to write and read data from JSON. In JSON, data is represented in key-value pairs, and curly braces hold objects, where a colon is followed after each name. The comma is used to separate key-value pairs. Square brackets are used to hold arrays, where each value is comma-separated.

#### Uses of JSON

- It is used while writing JavaScript based applications that includes browser extensions and websites.
- JSON format is used for serializing and transmitting structured data over network connection.
- It is primarily used to transmit data between a server and web applications.
- Web services and APIs use JSON format to provide public data.
- It can be used with modern programming languages.

#### Characteristics of JSON

- JSON is easy to read and write.
- It is a lightweight text-based interchange format.
- JSON is language independent.

### 4.3.5 Node.js

Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

**Advantages of NodeJS:** Here are the benefits of using Node.js

1. **Easy Scalability:** Developers prefer to use Node.js because it is easily scaling the application in both horizontal and vertical directions. We can also add extra resources during the scalability of the application.

2. **Real-time web apps:** If you are building a web app you can also use PHP, and it will take the same amount of time when you use Node.js, But if I am talking about building chat apps or gaming apps Node.js is much more preferable because of faster synchronization. Also, the event loop avoids HTTP overloaded for Node.js development.
3. **Fast Suite:** NodeJs runs on the V8 engine developed by Google. Event loop in NodeJs handles all asynchronous operation so NodeJs acts like a fast suite and all the operations can be done quickly like reading or writing in the database, network connection, or file system.
4. **Easy to learn and code:** NodeJs is easy to learn and code because it uses JavaScript. If you are a front-end developer and have a good grasp of JavaScript you can easily learn and build the application on NodeJS
5. **Advantage of Caching:** It provides the caching of a single module. Whenever there is any request for the first module, it gets cached in the application memory, so you don't need to re-execute the code.
6. **Data Streaming:** In NodeJs HTTP request and response are considered as two separate events. They are data stream so when you process a file at the time of loading it will reduce the overall time and will make it faster when the data is presented in the form of transmissions. It also allows you to stream audio and video files at lightning speed.
7. **Hosting:** PaaS (Platform as a Service) and Heroku are the hosting platforms for NodeJS application deployment which is easy to use without facing any issue.
8. **Corporate Support:** Most of the well-known companies like Walmart, Paypal, Microsoft, Yahoo are using NodeJS for building the applications. NodeJS uses JavaScript, so most of the companies are combining front-end and backend Teams together into a single unit.

### 4.3.6 MongoDB

MongoDB is an open-source document-oriented database that is designed to store a large scale of data and also allows you to work with that data very efficiently. It is categorized under the NoSQL (Not only SQL) database because the storage and retrieval of data in the MongoDB are not in the form of tables.

The MongoDB database is developed and managed by MongoDB.Inc under SSPL(Server Side Public License) and initially released in February 2009. It also provides official driver support for all the popular languages like C, C++, C#, and .Net, Go, Java, Node.js, Perl, PHP, Python, Motor, Ruby, Scala, Swift, Mongoid. So, that you can create an application using any of these languages. Nowadays there are so many companies that used MongoDB like Facebook, Nokia, eBay, Adobe, Google, etc. to store their large amount of data.

#### Major features of MongoDB

##### 1. Support ad hoc queries

In MongoDB, you can search by field, range query and it also supports regular expression searches.

##### 2. Indexing

You can index any field in a document.

##### 3. Replication

MongoDB supports Master Slave replication.

A master can perform Reads and Writes and a Slave copies data from the master and can only be used for reads or back up (not writes)

##### 4. Duplication of data

MongoDB can run over multiple servers. The data is duplicated to keep the system up and also keep its running condition in case of hardware failure.

##### 5. Load balancing

It has an automatic load balancing configuration because of data placed in shards.

##### 6. Supports map reduce and aggregation tools.



- 7. Uses JavaScript instead of Procedures.**
- 8. It is a schema-less database written in C++.**
- 9. Provides high performance.**
- 10. Stores files of any size easily without complicating your stack.**
- 11. Easy to administer in the case of failures.**
- 12. It also supports:**
  - JSON data model with dynamic schemas
  - Auto-sharing for horizontal scalability
  - Built in replication for high availability

## **Advantages of MongoDB**

- Schema less – MongoDB is a document database in which one collection holds different documents. Number of fields, content and size of the document can differ from one document to another.
- Structure of a single object is clear.
- No complex joins.
- Deep query-ability. MongoDB supports dynamic queries on documents using a document-based query language that's nearly as powerful as SQL.
- Tuning.
- Ease of scale-out – MongoDB is easy to scale.
- Conversion/mapping of application objects to database objects not needed.
- Uses internal memory for storing the (windowed) working set, enabling faster access of data.

## **5.CODING PAGES**

## 5.1 Main Page

```
import
'package:aswin/common/widgets/bottom_bar.dart'
; import
'package:aswin/constants/globalvariables.dart';
import
'package:aswin/features/admin/screen/admin_screen.dart';
import
'package:aswin/features/auth/screen/auth_screen.dart';
import 'package:aswin/provider/user_provider.dart';
import
'package:aswin/router.dart';
import
'package:flutter/material.dart';
import
'package:provider/provider.dart';

import 'features/auth/services/auth_service.dart';

void main() {
  runApp(MultiProvider(providers:
    [ ChangeNotifierProvider(
      create: (context) => UserProvider(),
    ),
    ], child: const MyApp()));
}

class MyApp extends
  StatefulWidget { const
  MyApp({super.key});
```

```

    @override
    State<MyApp> createState() => _MyAppState();
}

```

```

class _MyAppState extends
    State<MyApp> { final AuthService
    authService = AuthService();

    @override
    void initState()
    {
        super.initState
        ();
        authService.getUserData(context);
    }
}

```

```

@override
Widget build(BuildContext
    context) { return MaterialApp(
    debugShowCheckedModeBanner
    : false,title: 'Flutter Demo',
    theme: ThemeData(
        scaffoldBackgroundColor:
            GlobalVariables.backgroundColor,colorScheme:
            const ColorScheme.light(
                primary: GlobalVariables.greyBackgroundColor,
            ),
        appBarTheme: const
        AppBarTheme( elevation: 0,
            iconTheme:
            IconThemeData(color:
            Colors.black,

```

```

    ),
    ),
    ),
    onGenerateRoute: (settings) => generateRoute(settings),
    home: Provider.of<UserProvider>(context).user.token.isNotEmpty
        ? Provider.of<UserProvider>(context).user.type == 'user'
            ? const BottomBar()
            : const AdminScreen()
        : const AuthScreen(),
  );
}
}

```

## 5.2 Router page

```

import 'package:aswin/common/widgets/bottom_bar.dart';
import
'package:aswin/features/address/screens/address_screen.dart'
; import
'package:aswin/features/home/screens/category_deals_screen.dar
t'; import
'package:aswin/features/home/screens/home_screen.dart';
import
'package:aswin/features/order_details/screen/order_details_screen.d
art'; import
'package:aswin/features/product_details/screens/product_details_screen.
dart'; import
'package:aswin/features/search/screens/search_screen.dart';
import
'package:aswin/models/order.dart';
import
'package:aswin/models/product.dart';

```

```

import

'package:flutter/material.dart';

import
'features/admin/screen/add_product_screen.dart';
import 'features/auth/screen/auth_screen.dart';

Route<dynamic> generateRoute(RouteSettings
routeSettings) { switch (routeSettings.name) {
  case AuthScreen.routeName:
    return
      MaterialPageRoute(
        settings:
          routeSettings,
        builder: (_) => const AuthScreen(),
      );
  case HomeScreen.routeName:
    return
      MaterialPageRoute(
        settings:
          routeSettings,
        builder: (_) => const HomeScreen(),
      );
  case BottomBar.routeName:
    return
      MaterialPageRoute(
        settings:
          routeSettings,
        builder: (_) => const BottomBar(),
      );

```

## Case

**AddProductScreen.routeName**

```
e: return MaterialPageRoute(  
settings: routeSettings,  
builder: (_) => const AddProductScreen(),  
);
```

**case CategoryDealsScreen.routeName:**

```
var category = routeSettings.arguments as  
String; return MaterialPageRoute(  
settings: routeSettings,  
builder: (_) =>  
CategoryDealsScreen(  
category: category,  
),  
);
```

**case SearchScreen.routeName:**

```
var searchQuery = routeSettings.arguments  
as String; return MaterialPageRoute(  
settings: routeSettings,  
builder: (_) =>  
SearchScreen(  
searchQuery:  
searchQuery,  
),  
);
```

**case ProductDetailsScreen.routeName:**

```
var product = routeSettings.arguments as  
Product; return MaterialPageRoute(  
settings: routeSettings,  
builder: (_) =>
```

```

        ProductDetailsScreen(product:
            product,
        ),
    );
case AddressScreen.routeName:
    var totalAmount = routeSettings.arguments as String;

    return
        MaterialPageRoute(
            settings: routeSettings,
            builder: (_) =>
                AddressScreen(
                    totalAmount:
                        totalAmount,
                ),
        );
case OrderDetailsScreen.routeName:
    var order = routeSettings.arguments as Order;

    return
        MaterialPageRoute(
            settings:
                routeSettings,
            builder: (_) =>
                OrderDetailsScreen(order:
                    order,
                ),
        );
default:

```



```

return
MaterialPageRoute(
  settings: routeSettings,
  builder: (_) => const
  Scaffold(body: Center(
    child: Text('Screen does not exist'),
  ),
),
);
}
}

```

### 5.3 Account Screen Page

```
import 'package:flutter/material.dart';
```

```
import
```

```
'../../constants/globalvariables.da
```

```
rt';import
```

```
'../widgets/below_app_bar.dart';
```

```
import '../widgets/orders.dart';
```

```
import '../widgets/top_buttons.dart';
```

```
class AccountScreen extends
```

```
StatelessWidget { const
```

```
AccountScreen({super.key});
```

```
@override
```

```
Widget build(BuildContext
```

```
context) { return Scaffold(
```

```
  appBar: PreferredSize(
```

```

preferredSize: const
Size.fromHeight(50),child:
AppBar(
  flexibleSpace: Container(
    decoration: const
    BoxDecoration(
      gradient: GlobalVariables.appBarGradient,
    ),

    title: Row(
      mainAxisAlignment:
      MainAxisAlignment.spaceBetween,children: [
        Container(
          alignment:
          Alignment.topLeft,child:
          Image.asset(
            'assets/images/mizoraimage(
            2).png',width: 190,
            height: 95,
            color: Colors.black,
          ),
        ),

        Container(
          padding: const EdgeInsets.only(left: 15,
            right: 15),child: Row(children: const [
            Padding(
              padding: EdgeInsets.only(right: 15),
              child: Icon(Icons.notification_add_outlined),
            ),

```

```

        Icon(Icons.search),
      ],
    ),
    ],
  ),
),
),
body:
  Column(
    children:
      const [
        BelowAppBa
          r(),
        SizedBox(
          height: 10,
        ),
        TopButton
          s(),
        SizedBox(
          height: 20,

        Orders(),
      ],
    ),
  );
}
}

```

## 5.4 Address Screen page

```
import 'package:aswin/constants/utills.dart';
import
'package:aswin/features/address/services/address_services.dart';
import 'package:aswin/main.dart';
import
'package:aswin/provider/user_provider.dart';
import 'package:flutter/material.dart';
import
'package:loading_indicator/loading_indicator.da
rt';import 'package:pay/pay.dart';
import 'package:provider/provider.dart';

import
'../../common/widgets/custom_botton.dart';
import
'../../common/widgets/custom_text_field.dart';
import ' ../../constants/globlavariabls.dart';

class AddressScreen extends
StatefulWidget { static const String
routeName = '/address'; final String
totalAmount;
const AddressScreen({super.key, required this.totalAmount});

@override
State<AddressScreen> createState() => _AddressScreenState();
}

class _AddressScreenState extends State<AddressScreen> {
```

```

final TextEditingController flatBuildingController =
TextEditingController();final TextEditingController areaController
= TextEditingController();
final TextEditingController pincodeController = TextEditingController();

final TextEditingController cityController =
TextEditingController();final _addressFormKey =
GlobalKey<FormState>();

String addressToBeUsed = '';

List<PaymentItem> paymentItems = [];
final AddressServices addressServices =
AddressServices();

@override
void initState() { super.initState();
{

super.initState();
paymentItems.a
dd(
PaymentItem(
amount:
widget.totalAmount,
label: 'Total Amount',
status: PaymentItemStatus.final_price,
),
);
}

```

**@override**

**void dispose() {**

**super.dispose();**

**flatBuildingController.dispose();**

**areaController.dispose();**

**pincodeController.dispose();**

**cityController.dispose();**

**}**

**void onGooglePayResult(res) {**

**if (Provider.of<UserProvider>(context, listen: false)**

**.user**

**.address**

**.isEmpty) {**

**addressServices.saveUserAddress(**

**context: context, address: addressToBeUsed);**

**}**

**addressServices.placeOrd**

**er( context: context,**

**address: addressToBeUsed,**

**totalSum: double.parse(widget.totalAmount),**

**);**

**}**

**void payPressed(String**

**addressFromProvider) {**

**addressToBeUsed = "";**

**bool isForm =**

**flatBuildingController.text.isNotEmpty ||**

```

    areaController.text.isNotEmpty ||
    pincodeController.text.isNotEmpty ||
    cityController.text.isNotEmpty;

if (isForm) {
    if
        (_addressFormKey.currentState!.valid
        ate()) { addressToBeUsed =
            '${flatBuildingController.text},${areaController.text},${cityController.text}-${pincodeController.text}';
        } else {
            throw Exception('Please enter all the values!');
        }
    } else if
        (addressFromProvider.isNotEmpty)
        { addressToBeUsed =
            addressFromProvider;
        } else {
            showSnackBar(context, 'Error');
        }
    }

@override
Widget build(BuildContext context) {
    var address = context.watch<UserProvider>().user.address;

    return Scaffold(
        appBar:
        PreferredSize(
            preferredSize: const
            Size.fromHeight(60),child:
            AppBar(

```

```

flexibleSpace: Container(
  decoration: const
    BoxDecoration(
      gradient: GlobalVariables.appBarGradient,
    ),
  ),
),
),
),
),
body:
  SingleChildScrollView(
    child: Padding(
      padding: const
        EdgeInsets.all(8.0),child:
      Column(
        children: [
          if
            (address.isNotEm
              pty) Column(
                children:
                  [
                    Containe
                      r(
                        width: double.infinity,
                        decoration:
                          BoxDecoration(
                            border: Border.all(
                              color: Colors.black12,
                            ),
                          ),
                        child: Padding(
                          padding: const

```



```

EdgeInsets.all(8.0),child:
Text(
  address,
  style: const
    TextStyle(
      fontSize: 18,
    ),
  ),
),
const
  SizedBox(
    height: 20,
  ),
const Text(

  'OR',
  style:
    TextStyle(
      fontSize: 18,
    ),
  ),
const
  SizedBox(
    height: 20,
  ),
],
),

```

```

Form(
  key:
    _addressFormKey,
  child: Column(
    children: [
      CustomTextField(
        controller:
          flatBuildingController,
        hintText: 'Flat,House
          no,Budilding',
      ),
      const
        SizedBox(
          height: 10,
        ),
      CustomTextField(
        controller: areaController, hintText:
          'Area,Street'),const SizedBox(
          height: 10,
        ),
      CustomTextField(
        controller: pincodeController, hintText:
          'Pincode'),const SizedBox(
          height: 10,
        ),
      CustomTextField(
        controller:
          cityController,
        hintText:
          'Town/City',
      ),
    ],
  ),
)

```

};

```
    ),
  ),
  Container(
    color: Colors.white12,
    padding: const
    EdgeInsets.all(20), child:
    GooglePayButton(
      onPressed: () =>
      payPressed(address), width:
      double.infinity,
      type:
      GooglePayButtonType.order
      , height: 50,
      // ignore: deprecated_member_use
      paymentConfigurationAsset:
      'gp.json', onPaymentResult:
      onGooglePayResult,
      paymentItems: paymentItems,
      loadingIndicator: const
      LoadingIndicator(indicatorType:
      Indicator.ballRotate, ),),
    ),
  ],
),
),
),
);
}
```

}

## 5.5 Admin.js

```
const express=require("express");
const {Product} =
require("../models/product");const
adminRouter=express.Router();
const
admin=require("../middlewares/admin");
const
Order=require("../models/order");
// adding product
adminRouter.post('/admin/add-
product',admin,async(req,res)=>{ try{
  const{name,description,images,quantity,category,price}
  =req.body; let product=new Product({
    name,
    descriptio
    n,images,
    quantity,
    category,
    price,
  });

  product=await product.save();
  res.json(product);
}catch(e){
  res.status(500).json({error:e.mes
  sage});
}});
```

**// Get all the product**

```
adminRouter.get('/admin/get-  
products',admin,async(req,res)=>{ try{  
  const products=await  
  Product.find({});  
  res.json(products);  
}catch(e){  
  res.status(500).json({error:e.mes  
sage});  
}  
});
```

**// Delete the product**

```
adminRouter.post('/admin/delete-  
product',admin,async(req,res)=>{ try{  
  const{id}=req.body;  
  let product=await  
  Product.findByIdAndDelete(id);  
  res.json(product);  
  
}catch(e){  
  res.status(500).json({error:e.mes  
sage});  
}  
})
```

```

adminRouter.get('/admin/get-
orders',admin,async(req,res)=>{ try{
const orders=await
Order.find({});res.json(orders);
}catch(e){
res.status(500).json({error:e.mes
sage});

}
});
adminRouter.post('/admin/change-order-
status',admin,async(req,res)=>{ try{
const{id,status}=req.body;
let order=await
Order.findById(id);
order.status=status;
order=await
order.save();
res.json(order);
}catch(e){
res.status(500).json({error:e.mes
sage});
}
});

```

```

adminRouter.get("/admin/analytics", admin, async
(req, res) => { try {
const orders = await
Order.find({});let
totalEarnings = 0;

```

```

for (let i = 0; i < orders.length; i++) {
  for (let j = 0; j < orders[i].products.length;
    j++) { totalEarnings +=
      orders[i].products[j].quantity * orders[i].products[j].product.price;

    }
  }
}

// CATEGORY WISE ORDER FETCHING

let mobileEarnings = await fetchCategoryWiseProduct("Mobiles");
let essentialEarnings = await
fetchCategoryWiseProduct("Essentials"); let
applianceEarnings = await
fetchCategoryWiseProduct("Appliances"); let
booksEarnings = await
fetchCategoryWiseProduct("Books");
let fashionEarnings = await fetchCategoryWiseProduct("Fashion");

let earnings = {
  totalEarnings,
  mobileEarnings,
  essentialEarnings,
  applianceEarnings,
  booksEarnings,
  fashionEarnings,
};

res.json(earnings);
} catch (e) {
  res.status(500).json({ error: e.message});
}
});

```

```

async function
  fetchCategoryWiseProduct(category){ let
    earnings=0;
    let categoryOrders=await
      Order.find({
        'products.product.category':cate
        gory,
      });
    for(let i=0;i<categoryOrders.length;i++){
      for(let j=0;j<categoryOrders[i].products.length;j++){
        earnings+=categoryOrders[i].products[j].quantity*categoryOrders[i].products[j].produ
        ct.price;
      }
    }
    return earnings;
  }

  module.exports=adminRouter;

```



## 5.6 Admin.js Middleware

```
const jwt=require
('jsonwebtoken'); const
User=require("../models/use
r") const
admin=async(req,res,next)=
>{
  try{
    const token=req.header('x-
auth-token');if(!token)
    return res.status(401).json({msg:"no auth token, access denied"});

    const
    verified=jwt.verify(token,'passwordKey');
    if(!verified) return res
    .status(401)
    .json({mag:"Token verification failed
autheriaztin denied"});const user=await
    User.findById(verified.id);
    if(user.type=='user'||user.type=='seller'){
      return res.status(401).json({msg:"You are not an admin"});
    }
    req.user=verifie
    d.id;
    req.token=toke
    n; next();
  }catch(err){
    res.status(500).json({error:err.
    message});
  }}; module.exports=admin;
```

## **6.SYSTEM TESTING**

## 6.1 TESTING AND EVALUATION

Testing is a process of executing a program with the intent of finding an error. Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design and coding. Testing includes verifications of the basic logic of each program and verification that the entire system works properly. Testing demonstrates that software functions appear to be working according to specification. In addition, data collected as testing is conducted provided a good indication of software quality as a whole. The debugging process is the most unpredictable part of testing process. Testing begins at the module level and works towards the integration of the entire computer-based system testing and debugging are different activities, but any testing includes debugging strategy for software testing must accommodate low level tests that are necessary to verify that a small source code segment has been currently implemented as well as high-level tests that validate major system function, against customer requirements. No testing is complete without verification and validation part. The goals of verification and validation activities are to access and improve the quality of work products generated during the development and modification of the software. There are two types of verification: life cycle verification and formal verification. Life cycle verification is the process of determining the degree to which the products of the given phase of the development cycle fulfil the specification established during the prior process. Formal verification is the rigorous mathematical demonstration that source code confirms to its specifications. Validation is a process of evaluating software at the end of the software development process to determine compilation with the requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The primary objectives, when we test software are the following:

- Testing is a process of exceeding with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.
- A successful test is one uncovers undiscovered errors.

Thus, testing plays a very critical role in determining the reliability and efficiency of the software and hence is very important stage in software development. Tests are to be conducted on the software to evaluate its performance under a number of conditions.

Ideally, it should so at the level of each module and also when all of them are integrated to from the completed system. Software testing is done at different levels.

## **6.2 TESTING STRATEGIES**

A strategy for software testing integrates software test case design method in to a well-planned series of steps that result in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote a customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project progresses.

### **The general characteristics of software testing strategies are:**

- Testing begins at the component level and works “outward” toward the integration of the entire computer system.
- Different testing techniques are appropriate at different points in time.

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

A strategy must provide guidance for the practitioner and set of mild stones for the manager. Because the step on the test strategy occurs at a time when deadline pressure begins to rise, progress must be measurable and problem must surface as early as possible.

The software team’s approach to testing is defining a plan that describes an overall strategy and a procedure that defines specific testing steps and tests that will be conducted. In the proposed system, if the administrator makes any attempt to login to the application without entering his password, then the system will not allow the user to login to the application.

## **6.3 TESTING TECHNIQUES**

The various testing techniques are given below:

### **6.3.1 WHITE BOX TESTING**

White-box testing is also called as glass-box testing, is a test case design method that goes to the control structure of the procedural design to derive test cases. Using white box testing methods, the software engineer can derive test cases that,

- ✓ Guarantee that all independent paths within a module have been exercised at Least once.
- ✓ Exercise all logical decision on their true and false sides.
- ✓ Execute all loops at their boundaries and within their operational sides.
- ✓ Exercise internal data structure to ensure their validity.

White box testing was successfully conducted on our system. All independent paths within a module have been executed at least once and all logical decisions have been exercised on their true and false sides.

### **6.3.2 BLACK BOX TESTING**

Black-box testing is also called as behavioural testing, focuses on the functional requirement of the software. It is a complimentary approach that is likely uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories.

- ✓ Incorrect or missing functions.
- ✓ Interface errors.
- ✓ Error on data structures or external database access.
- ✓ Behaviour or performance errors.
- ✓ Initialization and termination errors.

Black box testing was successfully conducted on your system. The system was divided into a number of modules and testing was conducted on each module. We have tested the system for incorrect or missing functions, interface and performance errors.

### **6.3.3 UNIT TESTING**

Unit testing comprises the set of tests performed by an individual programmer prior to the integration of the system. Testing removes residual bugs and improves the reliability of the system.

Testing allows the developer to find out the design faults if any, and enable correction if needed. Exhaustive unit testing has to be carried out to ensure the validity of the data. In order to successfully test the entire package, unit testing is carried out. Each module was tested as when it was developed. Thus it proved easier to conduct minute testing operation and correct them then and there.

### **6.3.4 INTEGRATION TESTING**

Bottom-up integration is the traditional strategy used to integrate the component of a software system into a functional whole. Bottom-up integration consists of unit testing, followed by subsystem testing and followed by testing of the entire system. Unit testing has the goal of discovering errors in the individual parts of the system.

Parts are tested in isolation from one another in an artificial environment known as “Test Harness”, which consists of driver programs and data necessary to exercise the modules. Unit testing should be as exhaustive as possible to ensure that each representative case handled by each module has been tested. Unit testing is eased by a system structure that is composed of small loosely coupled modules.

Both control and data interfaces must be tested. Large software system may require several levels of subsystem testing. Lower level subsystems are successively combined to form higher level subsystems. In most software systems, exhaustive testing of subsystem capabilities is not feasible due to the combination complexity of the module interfaces. Therefore, test cases must be carefully chosen to exercise the interfaces in the desired manner.

### **6.3.5 ACCEPTANCE TESTING**

Acceptance testing involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements.

It is not unusual for two sets of acceptance test to be run, those developed by the quality group and those developed by the customer.

In addition to the functional and performance tests, stress tests are performed to determine the limitation of the system. For example, a compiler might be tested to determine the effect of the symbol table overflow, or real-time system might be tested to determine the effect of simultaneous arrival of numerous high priority interrupts.

### **6.3.6 OUTPUT TESTING**

Output testing of the proposed system is important since no system could be useful if it does not produce the required output.

The output format on the screen is found to be correct, as the format was designed in the system design phase according to the user needs. For the hard copy also the output comes out as the specified requirements by the user. Hence output testing doesn't result in any correction on the system.

# **7.SYSTEM IMPLEMENTATION AND DEPLOYMENT**



Implementation is the process of deploying the new system in the operational environment. Proper implementation is essential to provide a reliable system to meet the organizational requirement. There are four types of implementation methods. They are Direct Changeover, Phased Implementation, Parallel Run and Pilot Approach. The most commonly used implementation methods are Pilot Approach and Parallel Run.

The system which is developed as a mobile application hence the other functions as normal application, as usual some development technologies are used in the implementation of the project. The language we selected to program this software is dart/flutter. The reason we selected dart is that it is a simple and powerful language that especially developed to create mobile application.

Technologies used in the development of the software are:

- ✓ Programming language – dart
- ✓ DBMS – MongoDB server
- ✓ Development tool – Visual Studio Code
- ✓ Development platform – Windows 11

The front end is dart, html and back end is MongoDB Server and Node.js. The system developed on Visual Studio Code in Windows 11 operating system.

## **8.CONCLUSION**

The Mizora application has been developed for all given conditions and it is found working effectively under the all the circumstances that may arise in the real environment. The software has been developed to enhance productivity.

This system is user friendly and is well efficient to make easy consumer ,seller experience by increasing the feasibility of the local economy.The system is done with an insight into the necessary modifications that may be required in the future. Hence the system can be maintained successfully without much work.

## **8.1 FUTURE ENHANCEMENT**

As a future venture, it is suggested to make some changes to provide more services and to provide information at right time in right manner.

The current system is designed to make online shopping easier and enhance the availability of local products around us. It can help in improving the financial economy of the local environment.

In the future the system can be extended to many features including an offline mode , highly efficient AI driven search engine, advanced customer review filters. If any development is necessary in future, it can be done without affecting the design by adding additional modules to the system.

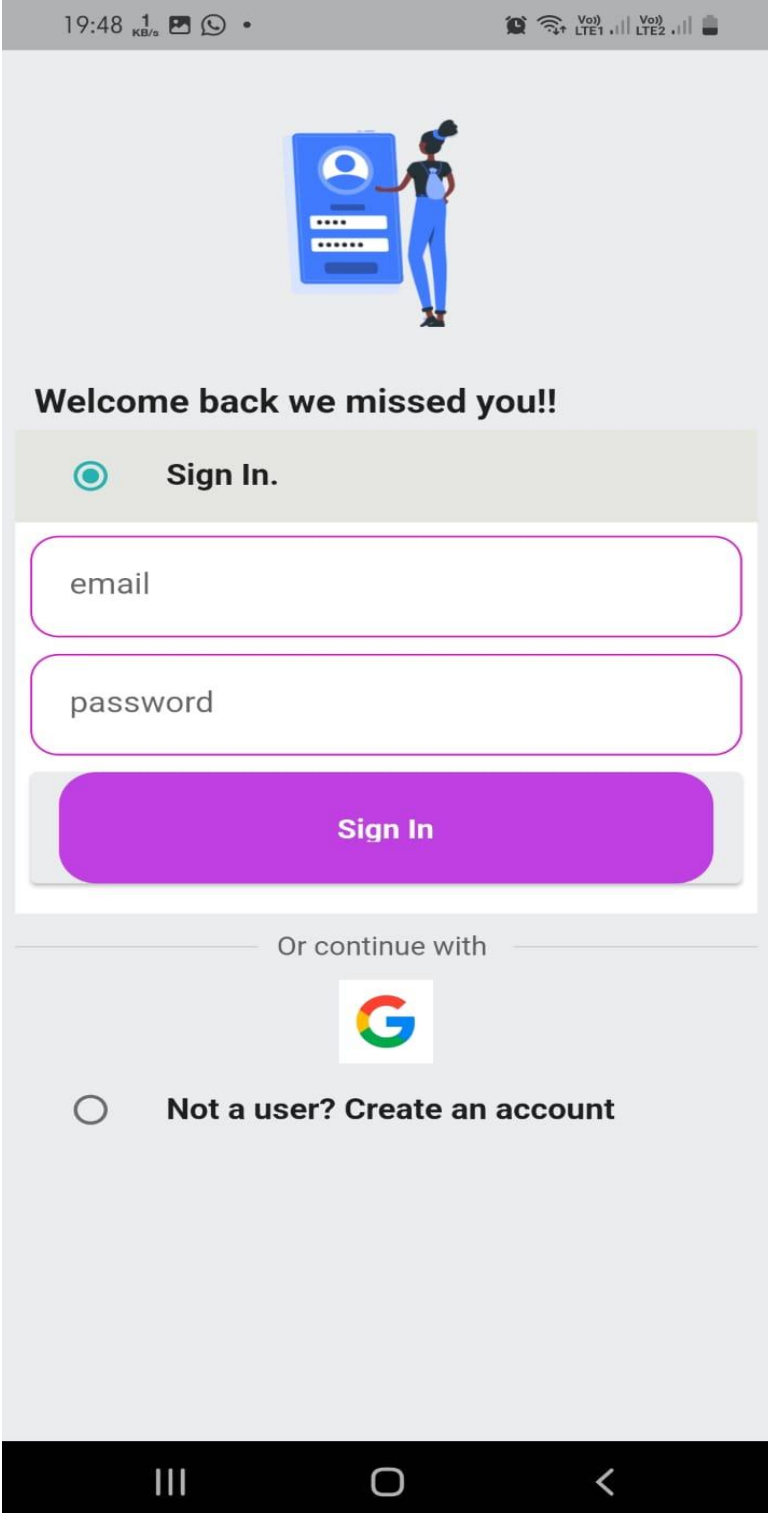
## **9. REFERENCE**

- Google.com
- YouTube


## **10.APPENDIX**

## 10.1 Screenshots

### 10.1.1 Login and Sign up



19:48 1 KB/s • VoLTE1 VoLTE2



**Welcome back we missed you!!**


☒ **Sign In.**

email

password

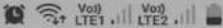
**Sign In**


Or continue with



☐ **Not a user? Create an account**

19:47 2 KB/s





### Join to our Family!!

☐ Sign In.

☒ Not a user? Create an account

name

email

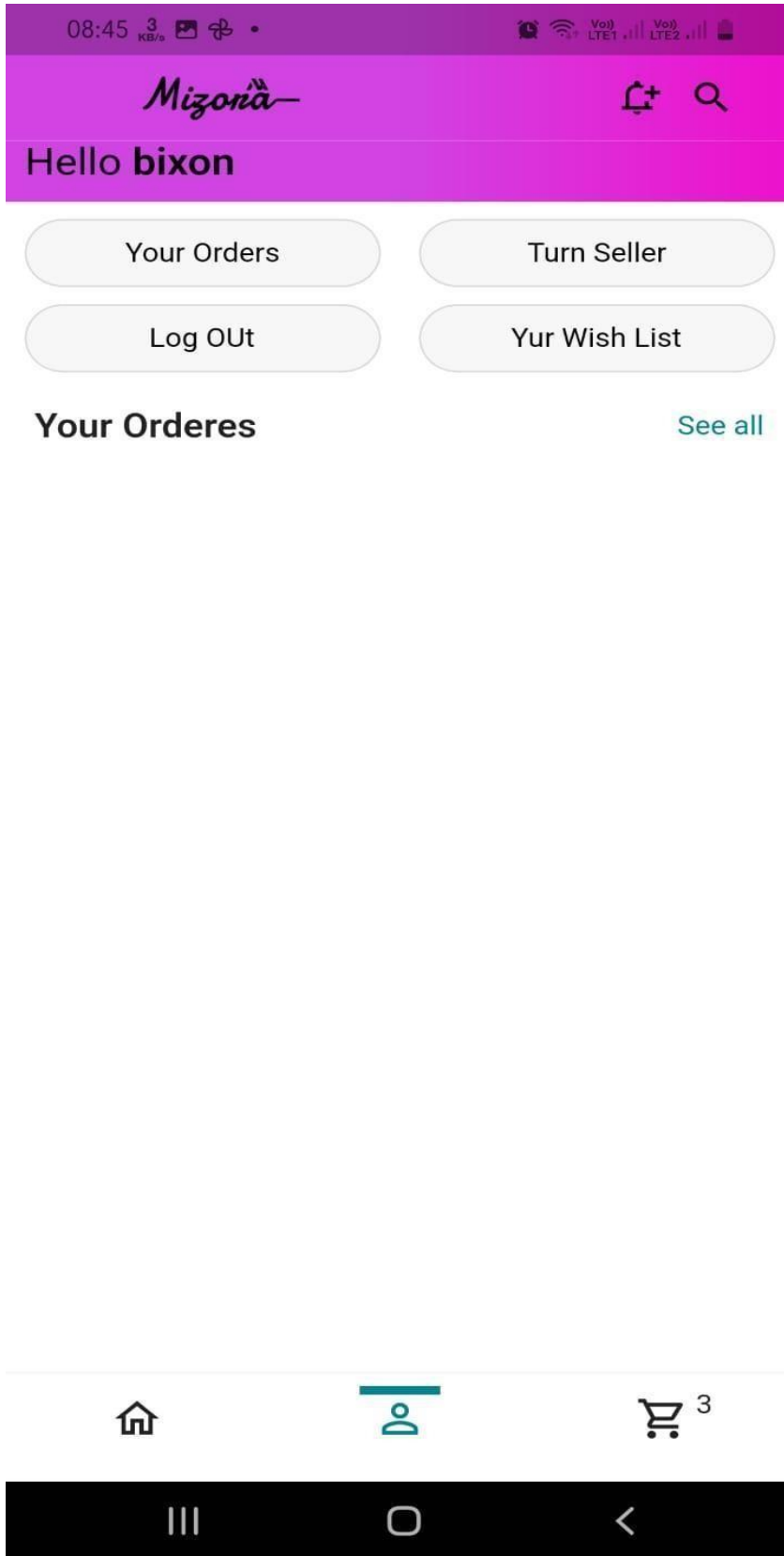
password

Sign Up

## User Home page







### 8.1.1 Adding Product

19:52 2 KB/s

← Add product

Select Product Images

MIZORA.IN

Product name

Description

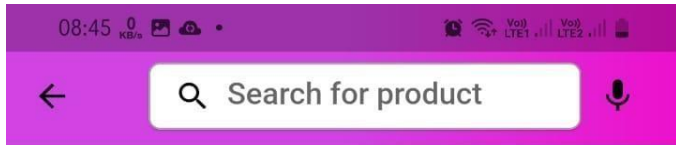
Price

Quantity

Localize ▾

Sell

## Product Page



### Washing Machine



Get it at: ₹68000.0

A washing machine (laundry machine, clothes washer, washer, or simply wash) is a home appliance used to wash laundry. The term is mostly applied to machines that use water as opposed to dry cleaning (which uses alternative cleaning fluids and is performed by specialist businesses) or ultrasonic cleaners. The user adds laundry detergent, which is sold in liquid or powder form, to the wash water.

Buy Now

Add to Cart

### Rate the Product



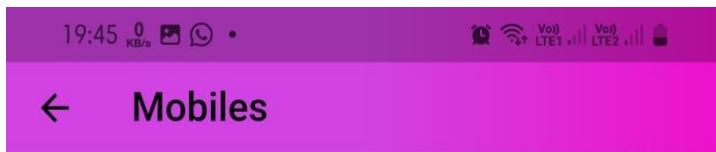
### Washing Machine



\$68000.0

Eligible for Free shipping

In Stock



keep looking for Mobiles



Galaxy s23



i phone



keep looking for Essentials



kitchen set

keep looking for Fashion



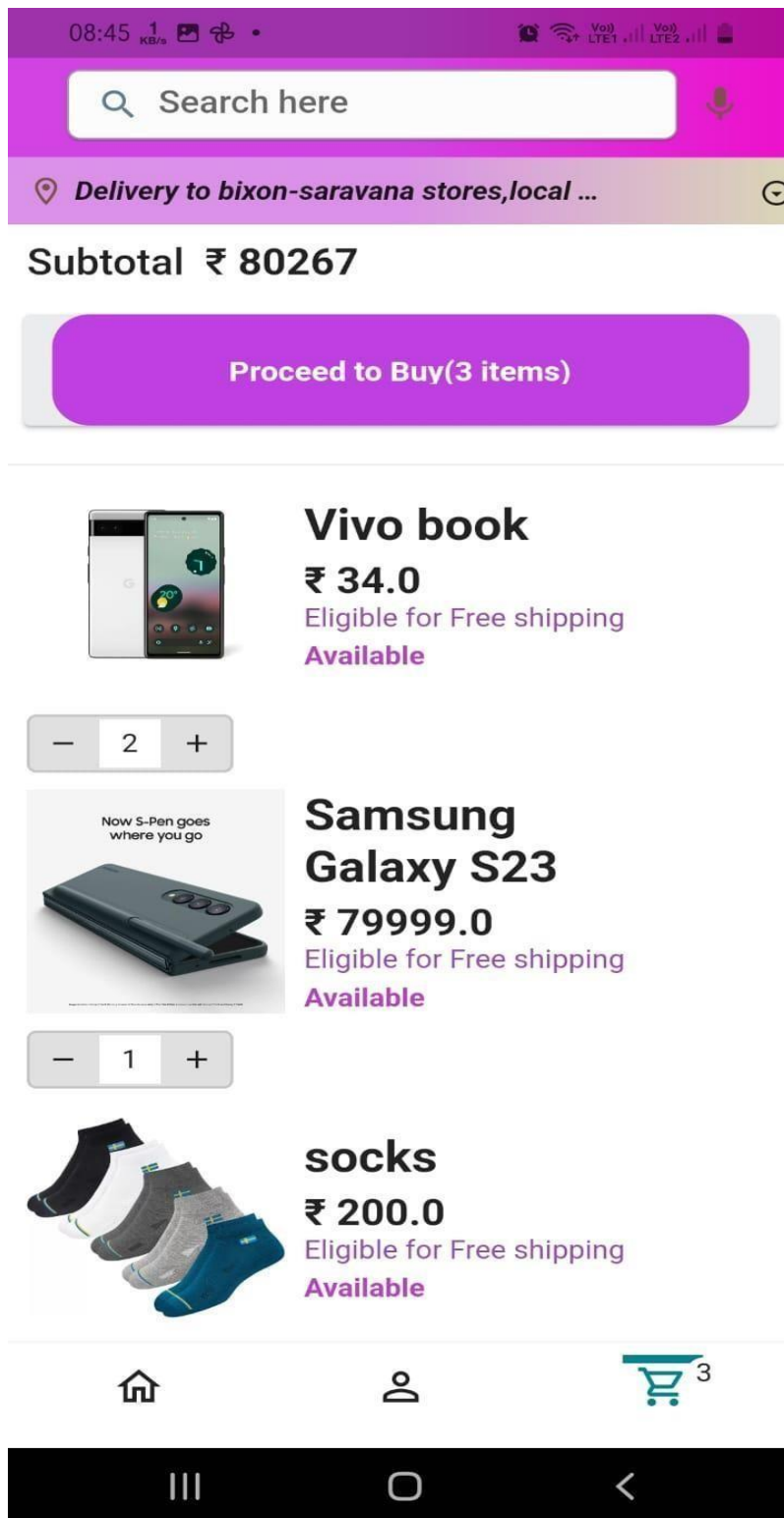
T-shirt for men



socks










## Cart Page





## Order page


9:03







 Search in Mizora



### View Order Details

Order Date:

January 10, 2023 1:12:29 PM


Order ID:

63bd16e51dc71cdb5180b196

Order Total:


\$ 23000.0


### Purchase Details




**Acer camera**  
Qty: 1

### Tracking

 Pending

 Completed  
Your order has been delivered ,You are yet to sign.  

Done

 Received



## Payment Page

08:46 2 KB/s •

←

saravana stores,local area,kannur-670703


OR

Flat,House no,Budilding

Area,Street

Pincode


Town/City


Order with 

||| ○ <

8:21 [Icons]



× Payment Details




 trollclub21@gmail.com ▼


+ Add new credit or debit card

Card number

# |  

 Scan card

MM/YY CVC

 Troll Club ✎

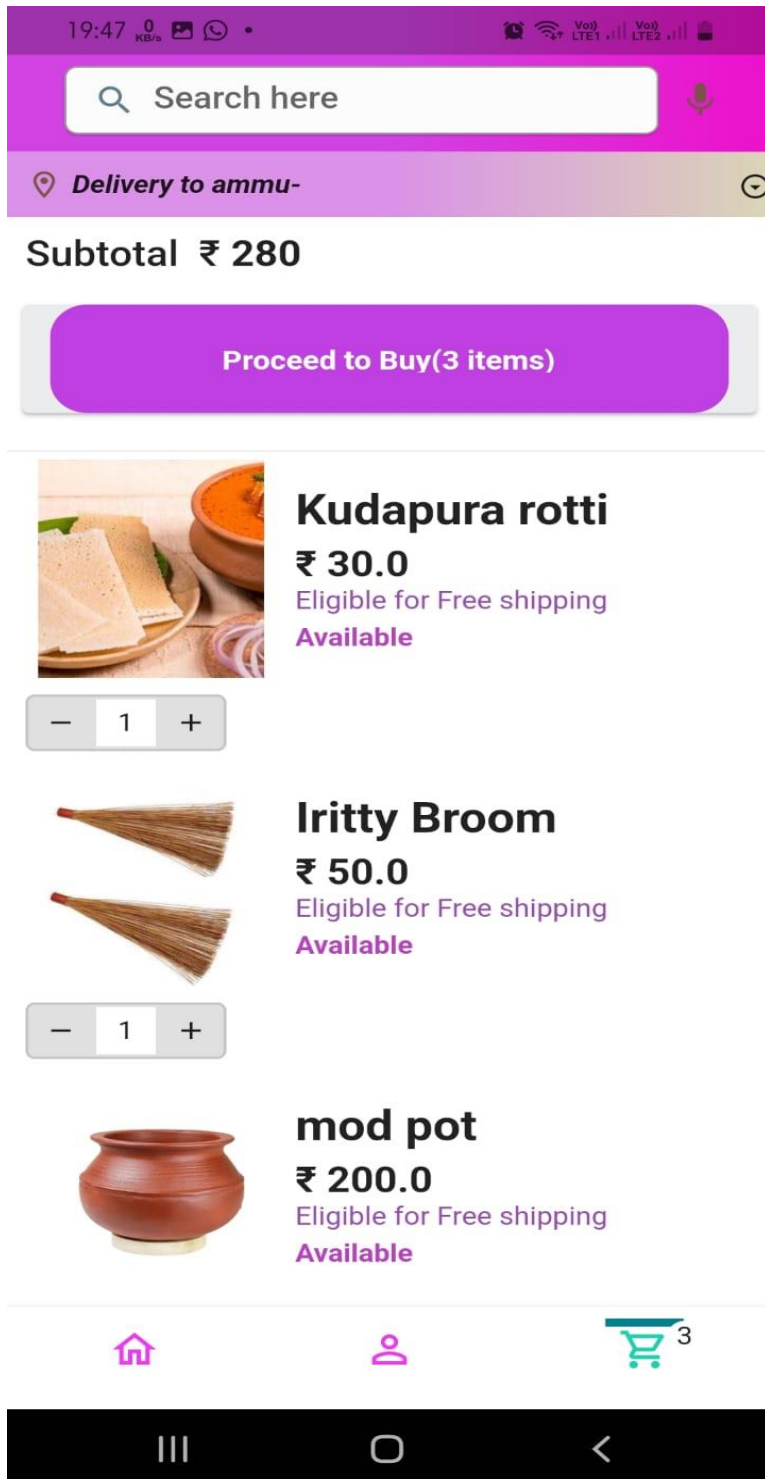
1 2 3 -

4 5 6 \_

7 8 9 ×

, 0 . →

## Localization



## ← Localize

keep looking for Localize



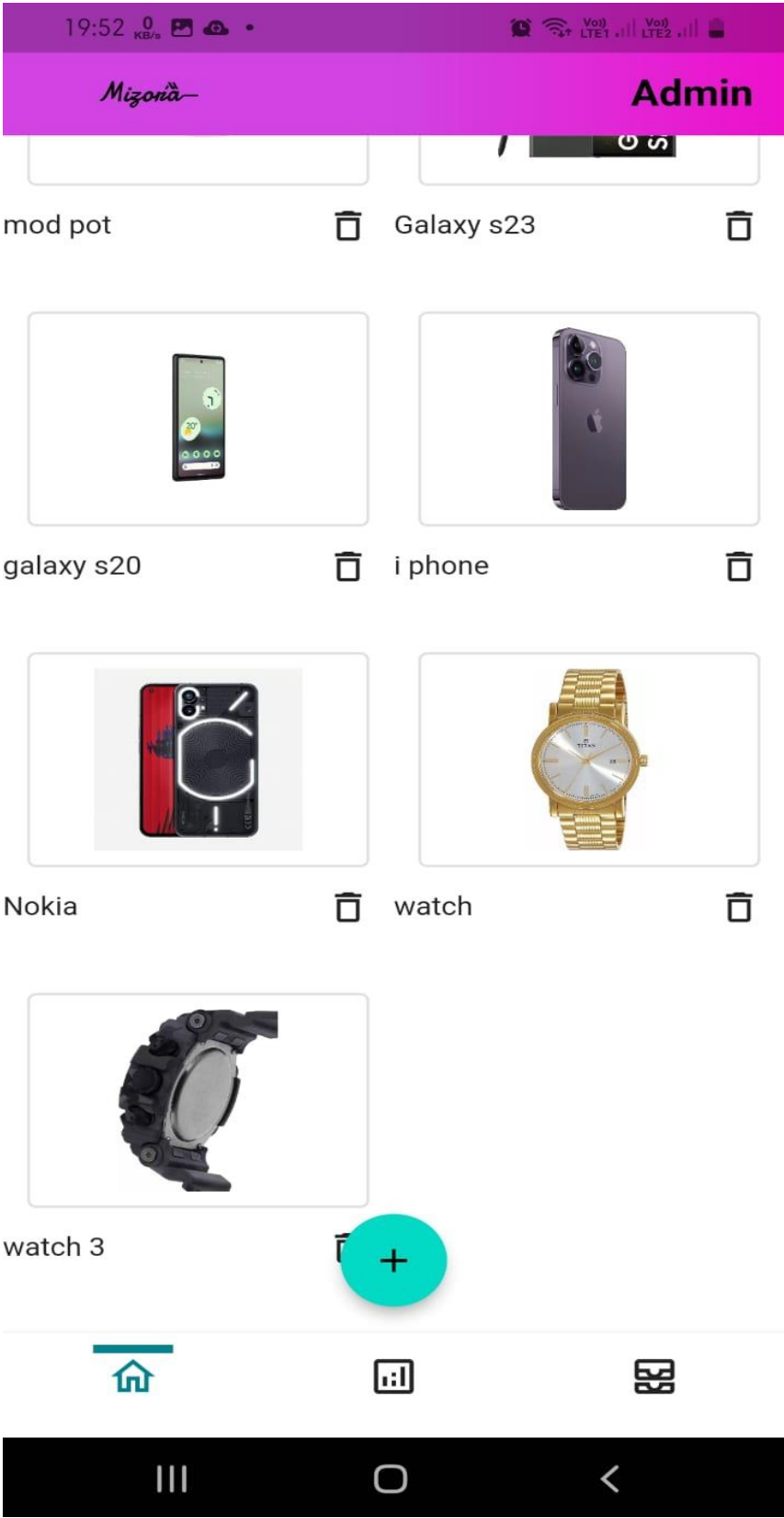
Kudapura rotti

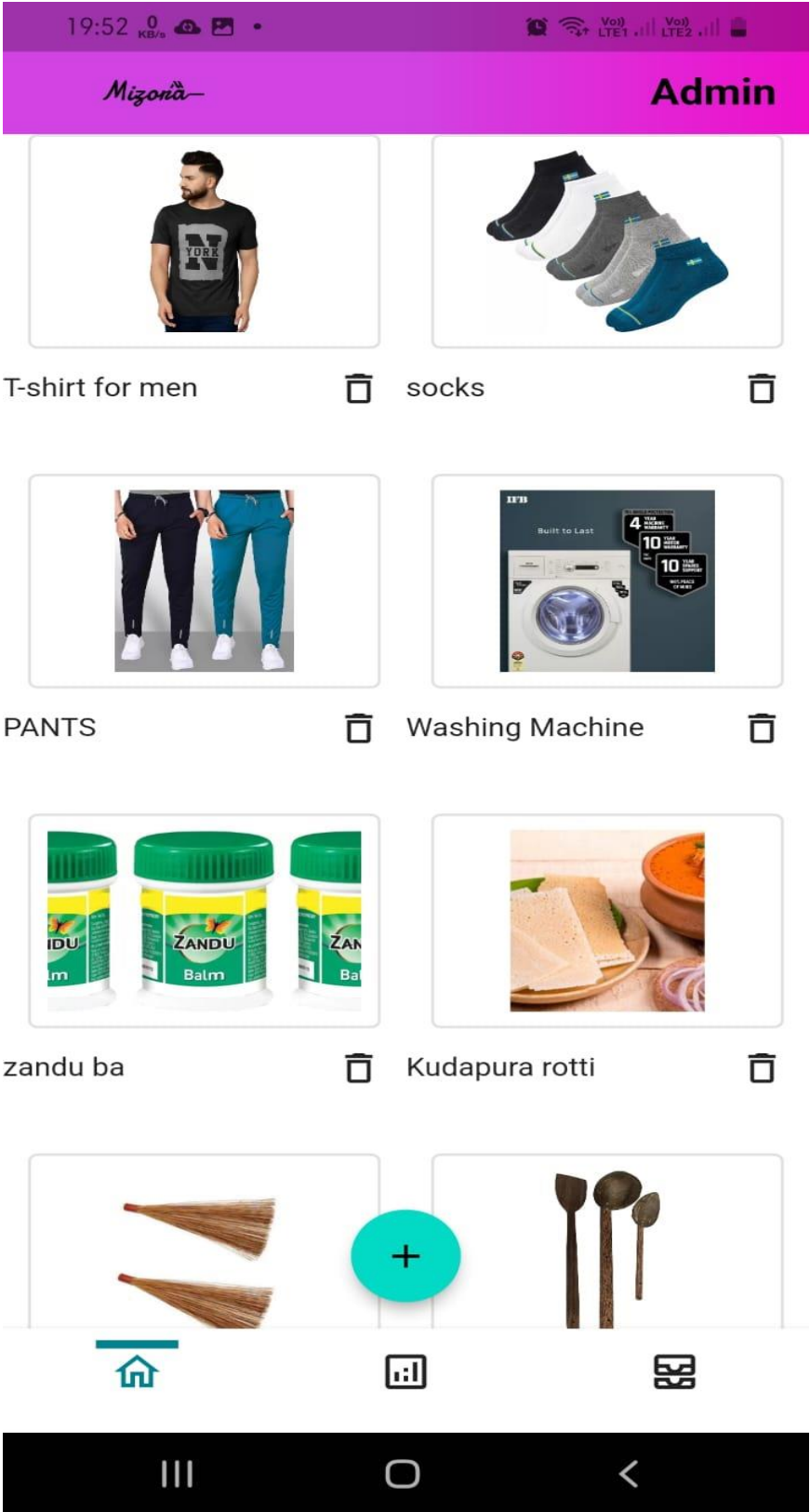


Iritty Broom



Product Admin Side





Sales Chart Page



**A PROJECT REPORT ON**  
**SAFE KERALA**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ASWIN SHIBU**

**REG.NO: DB20BCAR25**

**JILSON SAJI VARGHESE**

**REG.NO: DB20BCAR30**

**MUHAMMAD FASAL T V**

**REG.NO: DB20BCAR31**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**

# **A PROJECT REPORT ON SAFE KERALA**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ASWIN SHIBU**

**REG.NO: DB20BCAR25**

**JILSON SAJI VARGHESE**

**REG.NO: DB20BCAR30**

**MUHAMMAD FASAL T V**

**REG.NO: DB20BCAR31**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**



**DON BOSCO ARTS & SCIENCE COLLEGE**  
**ANGADIKADAVU**  
**IRITTY, KANNUR**



**CERTIFICATE**

*Certified that this report titled **SAFE KERALA-** is a bonafide record of the project work done by **Aswin Shibu(Reg.No: DB20BCAR25)** , **Muhammad Fasal T V (Reg.No:DB20BCAR31)** and **Jilson saji varghese(Reg.No:DB20BCAR30)** under the supervision and guidance ,towards partial fulfilment of the requirement for award of the degree of bachelor of computer application (BCA) of the Kannur university.*

Project Guide

Head of the Department

Angadikadavu

External Examiner

Date:

- 1.
- 2.

## DECLARATION

We ASWIN SHIBU and JILSON SAJI VARGHESE, MUHAMMAD FASAL T V sixth semester BCA student of Don Bosco Arts & Science College, Angadikadavu, under Kannur University do hereby declare that the project entitled “**SAFE KERALA**” is the original work carried out by me in the sixth semester under the supervision of Ms. Sruthi Nimesh, Lecturer of the Department of BCA, Don Bosco Arts & Science College, Angadikadavu, in partial fulfilment of the requirement for the award of the degree Bachelor of Computer Application, Kannur University.

Angadikadavu

Date:

ASWIN SHIBU

JILSON SAJI VARGHESE

MUHAMMAD FASAL T V

## **ACKNOWLEDGEMENT**

First of all we thank the lord almighty for his immense grace and blessings showered on me at every stages of this work. I am greatly indebted to our Principal Fr. Dr. Francis Karackat SDB, Don Bosco Arts & Science College, Angadikadavu for providing the opportunity to take up this project as part of my curriculum.

We deeply indebted to my project guide Ms. Sruthi N, lectures of department of BCA, for her assistance and valuable suggestions as guide. She made this project a reality.

We express our sincere thanks to Mrs. Sindu PM, Mr. Hebin Layola, Mrs. Fincy Cyriac and Mrs. Vineetha Mathew, lecturers of department of BCA, for their valuable suggestions during the course of this project. Their critical suggestions helped me to improve the project work.

Acknowledging the efforts of everyone, their chivalrous help in the course of the project preparation and their willingness to collaborate with the work, their magnanimity through lucid technical details lead to the successful completion of my project.

We would like to express my sincere thanks to all my friends, colleagues, parents and all those who have directly or indirectly assisted during this work.

# CONTENTS

<b>CHAPTE RS</b>	<b>CONTENTS</b>	<b>PAGE NO</b>
1	Introduction	1
2	System Analysis	7
2.1	Existing System	8
2.1.1	Disadvantage Of Existing System	8
2.2	Proposed System	8
2.2.1	Advantage Of Proposed System	8
2.3	Feasibility Study	9
2.3.1	Economic Feasibility	9
2.3.2	Technical Feasibility	10
2.3.3	Behavioural Feasibility	10
2.4	System Specification	10
2.4.1	Software Specification	10
2.4.2	Hardware Specification	11
2.5	Identification Of Actors	11
2.6	Identification Of Use Cases	12
2.6.1	Use Cases For The Actor Administrator	12
2.6.2	Use Cases For The Police station	13
2.6.3	Use Cases For The Labourer	13
2.6.4	Use Case Diagram	14

3	SYSTEM DESIGN	18
3.1	Introduction	19
3.2	Database Design	19
3.3	Table Design	20
3.4	Safe Kerala	21
3.5	Data Flow Diagram	26
3.6	ER Diagram	35
4	CODING	37
4.1	Input Interface	38
4.2	Output Interface	38
4.3	Software Description	38
4.4	HTML	38
4.4.1	CSS	39
4.4.2	JavaScript	40
4.4.3	MySQL	41
4.4.4	Python	42
4.4.5	Pycharm	46
4.4.6	Flask	46
5	Android studio	49
5.1	Coding pages	51
5.2	Admin pages	52
5.3	User page	72
6	System Testing	75

6.1	Testing And Evaluation	76
6.2	Testing Strategies	77
6.3	Testing Techniques	78
6.3.1	White Box Testing	78
6.3.2	Black Box Testing	78
6.3.3	Unit Testing	79
6.3.4	Integration Testing	79
6.3.5	Acceptance Testing	79
6.3.6	Output Testing	80
7	SYSTEM IMPLEMENTATION AND DEPLOYMENT	81
8	CONCLUSION	83
9	FUTURE ENHANCEMENT,REFERENCE	82,85
10	APPENDIX	86

## **1.INTRODUCTION**

## **1.1 Project Overview**

At present it is very difficult to find authorized workers from outside Kerala. Migrated Workers Portal is a solution for this difficulty. It is an interactive web enabled application. After identity verification police can add workers details to this site. So the users can easily search authorized workers with their needs. Police can block the workers from site if they are in criminal list. If the users have any complaint about worker they can inform to the police through this site. The main website feature is admin can monitor all functions and provide authorized workers for users. The workers can register to this site with their technical skills. So here the searching is based on the technical skills of workers.

## **NEED FOR THE SYSTEM**

Migrant labourers in Kerala, India's southernmost state, are a significant economic force in the state; there were around 2.5 million internal migrants in Kerala according to a 2013 study by the Gulati Institute of Finance and Taxation. Every year, the migrant worker population in Kerala increases by 2.35 lakh (235,000) people. The study, based on long-distance trains terminating in Kerala, does not cover migrants from the neighbouring states who use other modes of transport. Assuming that the estimation is rigorous and extrapolating it, taking into account the net annual addition, possible growth in migration rate, as well as accounting for the migration from the neighbouring states, Kerala is likely to have 5 to 5.5 million inter-state migrant workers in 2020. Despite their importance and despite many of them praising the state for its welfare schemes and environment, they are often ignored in comparison and suffer from comparatively poor living conditions. So safe Kerala helps us to find authorized and skilled migrated labourers.

## **OBJECTIVES AND SCOPE**

In this new era, as people get busier, the rate of violations also goes up. Though there are rules and regulations built to be followed, people mostly give no importance to it. Though there are police to check for violations, it makes it difficult for them to manually check every person for any violations. By introducing this website we aim to reduce the rate of violations among migrated labourers. The authorized government officer becomes the admin who monitors the system. Police station of various places



are assigned to enter the details of all the migrated labours. When any labour violates the rule they can be reported using the android version

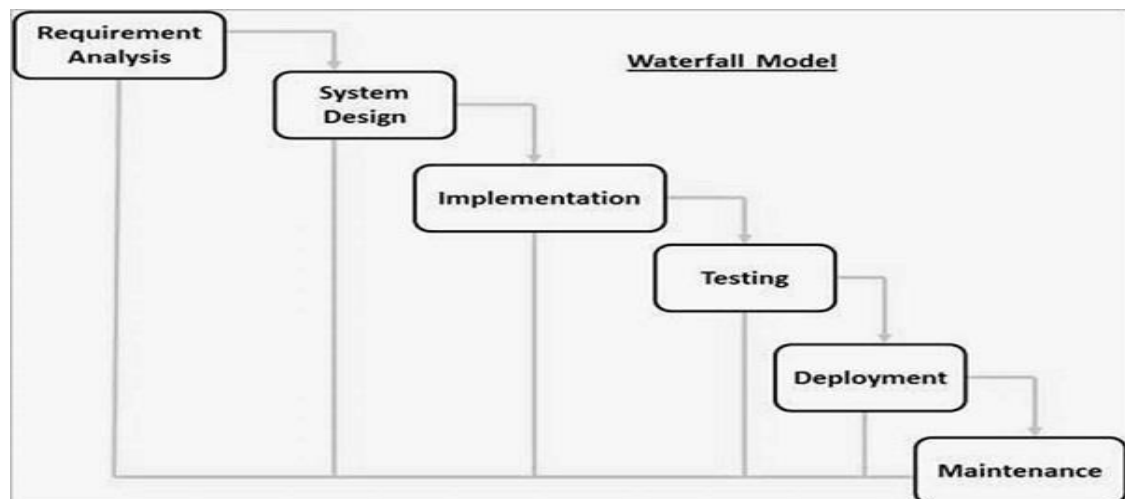
The scope of this project is high as it reduces manual work and utilises the resources in an efficient manner with minimum usage of time.

## MODEL:

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially

### Waterfall Model - Design:

In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. Following is the pictorial representation of Iterative and Incremental model:



The sequential phases in Waterfall model are –

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

### **Waterfall Model - Application:**

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are –

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.

- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

**The advantages of the waterfall model SDLC Model are as follows:**

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Some of the major advantages of the Waterfall Model are as follows –

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

**The disadvantages of the waterfall model SDLC Model are as follows:**

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows –

- No working software is produced until late during the life cycle.

- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

## **2. SYSTEM ANALYSIS**

## **INTRODUCTION**

At present it is very difficult to find authorized workers from outside Kerala. Migrated Workers Portal is a solution for this difficulty. It is an interactive web enabled application. After identity verification police can add workers details to this site. So the users can easily search authorized workers with their needs. Police can block the workers from site if they are in criminal list. If the users have any complaint about worker they can inform to the police through this site. The main website feature is admin can monitor all functions and provide authorized workers for users. The workers can register to this site with their technical skills. So here the searching is based on the technical skills of workers.

### **2.1 EXISTING SYSTEM**

For the successful designing of the management system it is necessary to study and analyze the present working mode of the existing system. Here it has been attained through the interview of the staffs using the existing system as well as the related officials. The existing system contains all the work done manually. All the details regarding the resource allocation are stored in separate register. The provisions for the citizens include registering a complaint. Different files are maintained for registering complaint, with drawing complaint, and also citizen list and to acquire other details. If a police reports a crime the details regarding the crime are kept in a file. Citizen details have to be maintained using separate files. The address of the citizen along with his all details is to be recorded. The retrieval of these data at times of emergencies is very tough task. Maintenance of proper records and other files are necessary in this site.

### **2.2 PROPOSED SYSTEM**

In the proposed system we proposed to computerize the above mentioned activities. In the existing system, all data processing is done manually. All the files and record books are replaced by the system software. When there are a lot of issues such as retrieval and storage of the information, reporting etc. And keeping track of them becomes a tedious task. By implementing a computerized system, the limitation in the present system will be reduced. Man power can be reduced to a great extent and efficiency and accuracy can be increased to manifold. More over consumption of time can be reduced to far greater extent by the implementation of the proposed system.

## **2.3 Feasibility Study**

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioral feasibilities.

The proposed system is theoretically investigated to check the feasibility and found that they are more reliable and efficient in the cases given below. There are three aspects in the feasibility study portion of the preliminary investigation.

✓ Economic feasibility

✓ Technical feasibility

✓ Behavioural feasibility

The proposed system must be evaluated from a technical point of view first, and if technical feasible their impact on the organization must be assessed. If compatible, the operational system can be devised. Then they must be tested for economic feasibility.

### **2.3.1 Economic Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors which affect the development of a new system is the cost it would require. Since the system developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

### **2.3.2 Technical Feasibility**

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, procedures and staff. Having identified an outline system, the investigation must go on suggest the type of equipment, required method developing the system, of running the system once it has been designed. The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Though the technology become obsolete after some period of time, due to the fact that newer version of some software supports older versions, the system still be used. So there are only minimal constraints involved with this project. The system has been developed using C# and .NET, along with the database software SQL server, thus we could conclude that the project is technically feasible for development.

### **2.3.3 Behavioural Feasibility**

People are inherently resistant to change and computers have been known to facilitate change. The System is designed in user friendly manner and we need to provide any special training for the persons using this software. The operating system used is Windows 10, which is also user friendly. Since the application is web biased and can easily accessed in a web browser, which is quite familiar to the intended users, it does not have any operational barriers. So no need to provide any special training for using this application software and hence it is behaviourally feasible.

## **2.4 System Specifications**

System Specification deals with the technical aspects the project has to meet in minimum to work successfully. This also includes the different aspects the software requirement is determined from. The technical details typically include:

- Software Specification
- Hardware Specification

### **2.4.1 Software Specifications**

The software required for the application depends on the following factors:

- ✓ The flexibility of the software



- ✓ Software contracts
- ✓ Limitation of the software

### **Software Requirement**

This specifies the minimum software requirements for implementing the system. This includes:

- Front End: - HTML, CSS, Java ,XML
- End Back: - Python 3.9, SQL
- Server side scripting: Python
- IDE: -DreamWeaver
- Operating System: Microsoft windows 7/8 for better performance

### **2.4.2 Hardware Specifications**

The software required for the application depends on the following factors:

- ✓ Determining size and capacity requirements.
- ✓ Computer evaluation and measurement.
- ✓ Financial factors.
- ✓ Maintenance and support.

### **Hardware Requirement**

- Processor: Intel Pentinum Core i3 and above
- Ram: 1 GB and above
- Hard disk: 40 GB and above

## **2.5 Identification of Actors**

A use cases represents the functionality of an actor. It is defined as a set of actions performed by a system, which yields an observable result. An ellipse containing its name inside the ellipse or below it represents. it. It is placed inside the system boundary and connected to an actor with an association. This shoes how the use cases and the actor interact. We can identify the actors through a list of questions. The answers to these questions bring out the actors of the system is.

- Admin
- Police station

- Labourer
- User

Here we need to specify the use cases of each actor.

## **2.6 Identification of use cases**

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yield an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

To find out the use cases, ask the following questions to each of the actors.

- ✓ Which functions does the actor require from the system? What does the actor need to do?
- ✓ Does the actor need to read, create, destroy, modify or store some kind of information in the system?
- ✓ Does the actor have to calculate something? And want to provide information for others?
- ✓ Could the actor's daily work be simplified or made more efficient by adding new functions to the system (typically functions which are currently not automated in the system)?

### **2.6.1 Use cases for the actor Administrator**

#### **Functions of Administrator:**

Administrator has the power manage all police stations. Admin can do the following functions;

- Login
- Police station management
- Work category management
- View criminal list
- View migrated labours
- View complaints and status
- Send notifications
- View feedback

### **2.6.2 Use cases for the actor Police station**

#### **Functions Police station**

Main functions include this module are;

- Login2.
- Labour management(ID card generate)
- Criminal history management
- View user complaints
- Send reply
- View feedback
- View notification

### **2.6.3 Use cases for the actor Labourer**

Main functions include this module are;

- Login
- View profile
- Add skill
- View booking
- Add bill
- View payment
- View previous work
- View reviews

### **2.6.4 Use cases for the actor user**

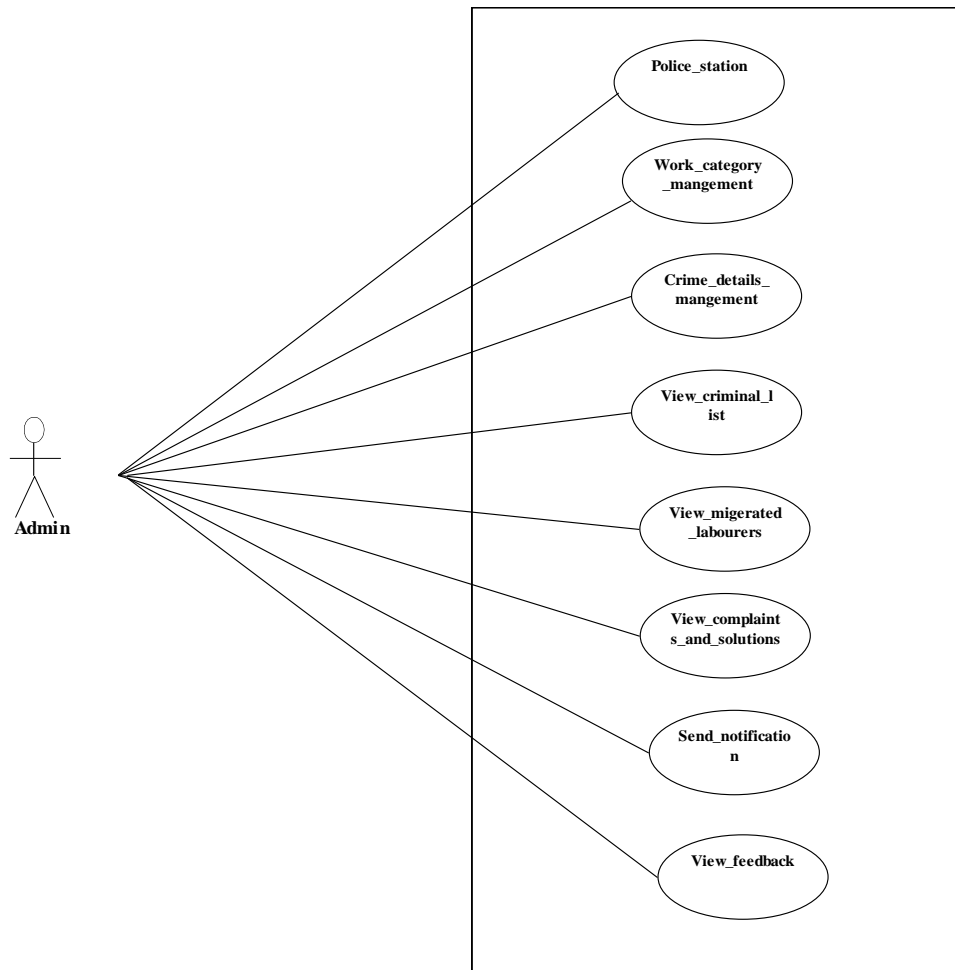
User has the power to register the complaint and also to view status of the complaint.

They can view solutions and missing notifications.

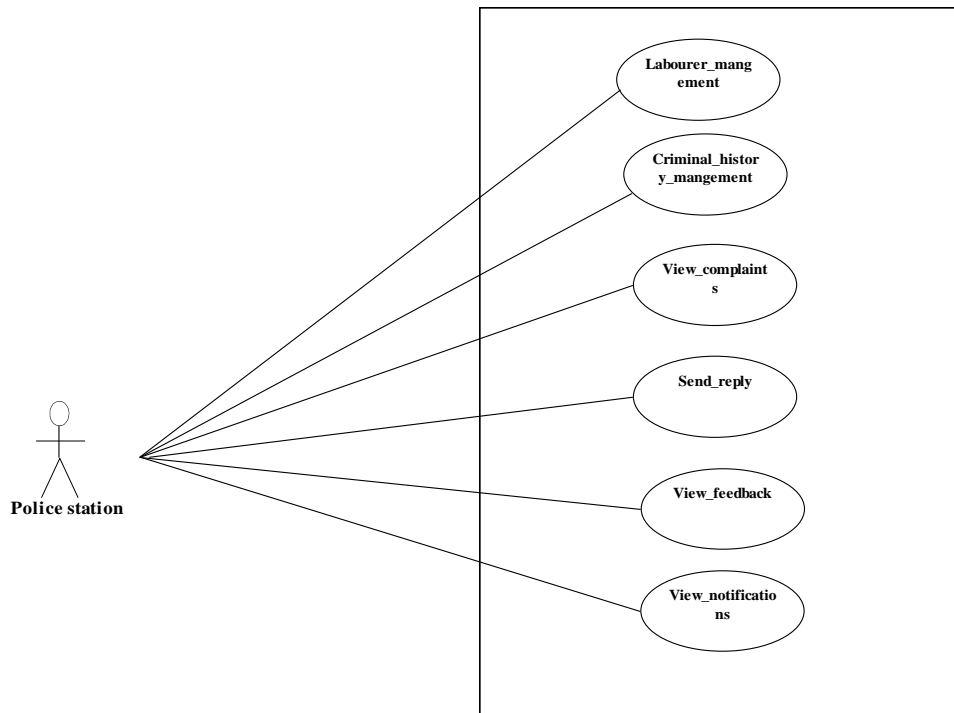
- Register
- Login
- View profile
- View work category
- View labourers
- Scan QR (ID card)
- Book worker
- View bill
- Payment
- Send review (about worker)
- View reviews

## 2.6.7 USE CASE DIAGRAM

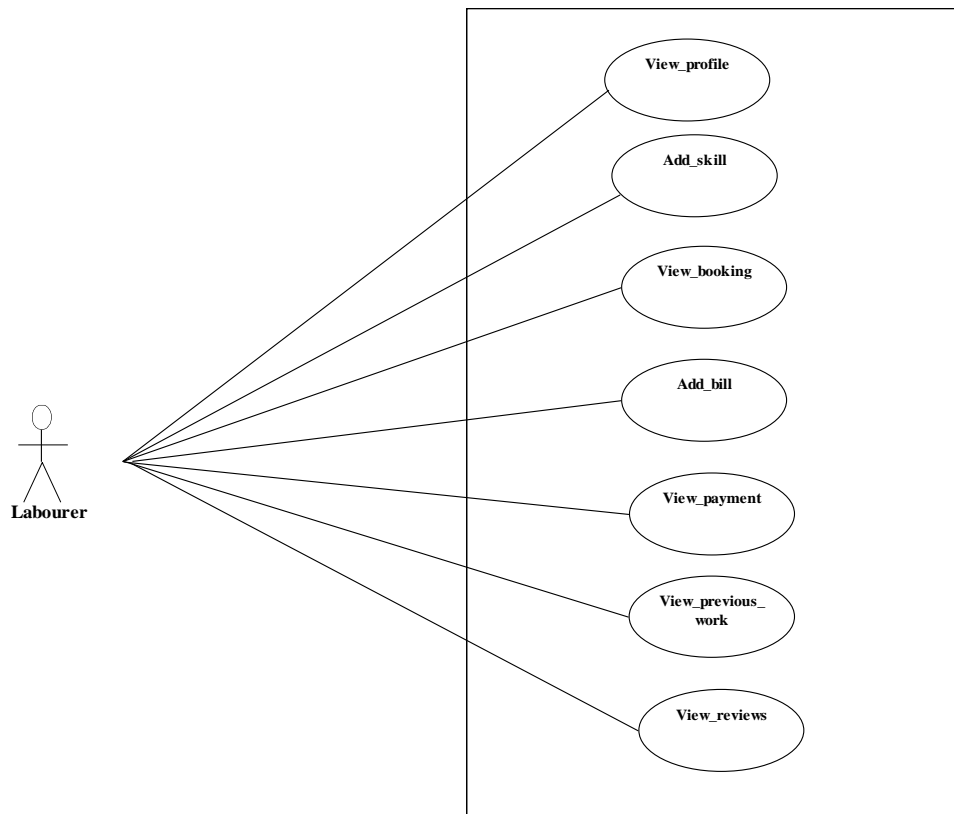
*ADMIN*

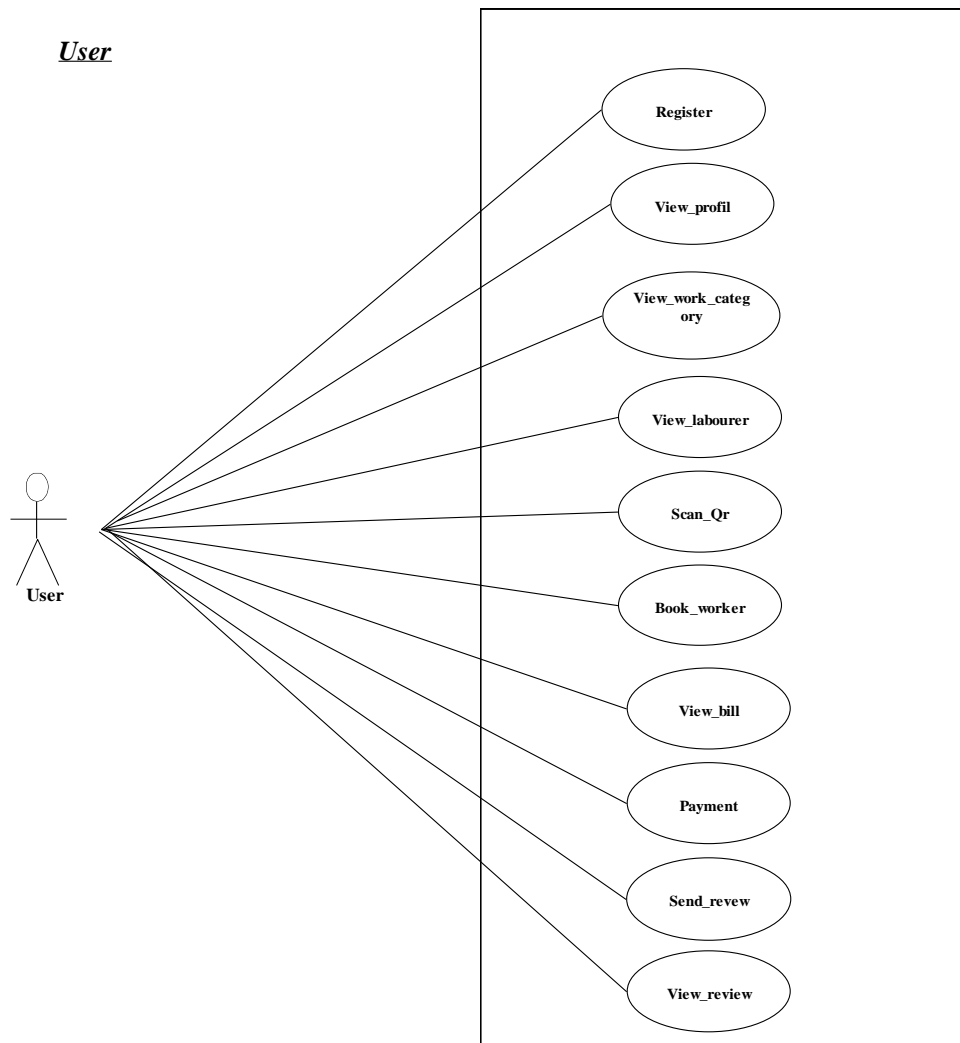


*police station*



**labourer**





### **3. SYSTEM DESIGN**



### **3.1 INTRODUCTION:**

System design provides an understanding of the procedural details, necessary implementing the system recommended in the feasibility study. Basically it is all about the creation of a new system. This is a critical phase since it decides the quality of the system and has a major impact on the testing and implementation phases.

**System design consists of three major steps.**

- Drawing of the expanded system data flow charts to identify all the processing functions required.
- The allocation of the equipment and the software to be used.
- The identification of the test requirements for the system.

### **CHARACTERS OF DESIGN**

- A design should exhibit a hierarchical organization that makes intelligent use of control among components of the software.
- A design should be modular that is, the software should be logical.
- A design should contain distinct and separable representation of data and procedure.
- A design should lead to interface that reduce the complexity of the connections between modules and with the external environment.

### **3.2 Database Design**

A Database is a collection of inter related data stored with minimum redundancy to serve many users quickly and efficiently. In database design data independence, accuracy, privacy and security are given higher priority. Database design is an integrated approach to the file design. This activity deals with the design of the physical data base. All entities and attributes have been identified while creating the database. The database design deals with the grouping of data into number of tables so as to.

- ✓ Reduplication of data.
- ✓ Minimize storage space.
- ✓ Retrieve the data efficiently.

Following are some guidelines for the database design:

- Design a relational schema so that it is easy to explain its meaning. Do not combine attributes from multiple entry and relationship type into a single relation.
- Design the database schema so that no insertion, deletion or modification anomalies are present in the relation.
- As far as possible, avoid placing attributes in the base relation whose values may frequently be null.
- Design relation schema so that they can be joined with equality conditions on attributes that are either primary keys or foreign keys in a way that no spurious tuples are generated.

### **3.3 Table Design**

DB design is required to manage large bodies of information. The management of data involves both the definition of the structure of storage of information and provisions of mechanism for the manipulation of information. For developing an efficient database certain conditions have to be fulfilled such as:

- Control Redundancy
- Ease of Use
- Data Independence
- Accuracy and Integrity

There are five major steps in design process:

- Identify the table and relationship.
- Identify the data that is needed for each table and relationship.
- Resolve the relationship.
- Verify the design.
- Implement the design

The Database Consist of the following tables given below.

### 3.4 Safe Kerala

#### 1.Login table

Column name	Data type	Constraints	Description
login_id	Int	Primary key	Unique identifier
Username	varchar(200)	Not null	Name of user
Password	varchar(200)	Not null	Secret key
Usertype	varchar(200)	Not null	To specify the type of user

#### 2.police station

<i>Column name</i>	<i>Data type</i>	<i>constraints</i>	<i>description</i>
<i>station_id</i>	<i>int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>station_name</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Name of station</i>
<i>email</i>	<i>varchar(200)</i>	<i>Not null</i>	
<i>district</i>	<i>varchar(200)</i>	<i>Not null</i>	
<i>ph</i>	<i>bgint(200)</i>	<i>Not null</i>	
<i>latitude</i>	<i>varchar(200)</i>	<i>Not null</i>	
<i>longitude</i>	<i>varchar(200)</i>	<i>Not null</i>	

#### 3. work category

Column name	Data type	Constraints	Description
category_id	Int	<i>Primary key</i>	Unique identifier
work_name	varchar(200)	<i>Not null</i>	Name of work

#### 4.crime details

<i>Column name</i>	<i>Data type</i>	<i>Constraints</i>	<i>Description</i>
<i>crime_id</i>	<i>Int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>crime_type</i>	varchar(200)	<i>Not null</i>	<i>Type of crime</i>
<i>crime_description</i>	varchar(200)	<i>Not null</i>	<i>About the crime</i>
<i>crime_date</i>	varchar(200)	<i>Not null</i>	<i>Date of crime</i>

## 5. Criminals

<i>Column name</i>	<i>Data type</i>	<i>Constraints</i>	<i>Description</i>
<i>criminal_id</i>	<i>Int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>station_id</i>	<i>Int</i>	<i>Foreign key</i>	<i>Id of station</i>
<i>labour_id</i>	<i>Int</i>	<i>Foreign key</i>	<i>Id of labour</i>
<i>crime_id</i>	<i>Int</i>	<i>Foreign key</i>	<i>Id of crime</i>
<i>crime_date</i>	<i>Int</i>	<i>Foreign key</i>	<i>Date of crime</i>
<i>Date</i>	<i>Int</i>	<i>Foreign key</i>	<i>Date</i>

## 6. Labourer

<i>Column name</i>	<i>Data type</i>	<i>Constraints</i>	<i>Description</i>
<i>Labourer_id</i>	<i>Int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>Labourer_name</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Labourers name</i>
<i>Labourer_dob</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Labourers dob</i>
<i>Labourer_details</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Labourer details</i>
<i>Labourer_photo</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Labourer photo</i>
<i>Labourer_phone_number</i>	<i>Int</i>	<i>Not null</i>	<i>Phone number</i>
<i>Labourer_email</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Labourers email</i>
<i>Labourer_gender</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Labourers gender</i>
<i>Proper_place</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Proper place</i>
<i>Proper_house</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Proper house</i>
<i>Proper_post</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Proper post</i>
<i>current_place</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Current place</i>
<i>Current_house</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Current house</i>
<i>Current_post</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Current post</i>
<i>Aadhar_number</i>	<i>bigint(20)</i>	<i>Not null</i>	<i>Aadhar number</i>

### 7. Complaints And Solution

Column name	Data type	Constraints	Description
complaint_id	Int	<i>Primary key</i>	Unique identifier
c_date	Date	<i>Not null</i>	Date of crime
user_id	Int	Foreign key	Id of user
complaints	varchar(200)	<i>Not null</i>	Complaints
Reply	varchar(200)	<i>Not null</i>	Reply
r_date	Date	<i>Not null</i>	Reply date

### 8. Notification

Column name	Data type	Constraints	Description
<i>notification_id</i>	<i>Int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>notification_message</i>	<i>varchar(200)</i>	<i>Not null</i>	Message
<i>date</i>	<i>varchar(200)</i>	<i>Not null</i>	Date

### 9. Bank

<i>Column name</i>	<i>Data type</i>	<i>Constraints</i>	<i>Description</i>
<i>bank_id</i>	<i>Int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>bank_name</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Bank name</i>
<i>ac_no</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Account number</i>
<i>ifsc</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Ifsc code</i>
<i>balance</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Bank balance</i>
<i>user_id</i>	<i>Int</i>	<i>Foregin key</i>	<i>Id of user</i>

### 10.Bill

<i>Column name</i>	<i>Data type</i>	<i>Constraints</i>	<i>Description</i>
<i>bill_id</i>	<i>int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>booking_id</i>	<i>int</i>	<i>Not null</i>	<i>Id of booking</i>
<i>bill_amount</i>	<i>int</i>	<i>Not null</i>	<i>Bill amount</i>
<i>bill_status</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Status of bill</i>

### 11.Booking

<i>Column name</i>	<i>Data type</i>	<i>Constraints</i>	<i>Description</i>
<i>booking_id</i>	<i>Int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>skill_id</i>	<i>Int</i>	<i>Foregin key</i>	<i>Id of skill</i>
<i>user_id</i>	<i>Int</i>	<i>Foregin key</i>	<i>Id of user</i>
<i>work_date</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Date of work</i>
<i>work_details</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Work details</i>
<i>booking_date</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Booking date</i>
<i>booking_time</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Booking time</i>
<i>booking_status</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Booking status</i>

### 12.Complaints

<i>Column name</i>	<i>Data type</i>	<i>Constraints</i>	<i>Description</i>
<i>complaint_id</i>	<i>int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>complaints</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>complaints</i>
<i>date</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Date</i>
<i>user_id</i>	<i>int</i>	<i>Foregin key</i>	<i>Id of user</i>
<i>reply</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Reply</i>
<i>station_id</i>	<i>int</i>	<i>Foregin key</i>	<i>Id of station</i>

### 13.Feedback

<i>Column name</i>	<i>Data type</i>	<i>Constraints</i>	<i>Description</i>
<i>feedback_id</i>	<i>Int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>user_id</i>	<i>Int</i>	<i>Foregin key</i>	<i>Id of user</i>
<i>date</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Date</i>
<i>feedback</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Feedback</i>

#### 14.Review

<i>Column name</i>	<i>Data type</i>	<i>Constraints</i>	<i>Description</i>
<i>review_id</i>	<i>Int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>review_date</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Date of review</i>
<i>user_id</i>	<i>Int</i>	<i>Foregin key</i>	<i>Id of user</i>
<i>review</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Review</i>
<i>lbid</i>	<i>Int</i>	<i>Foregin key</i>	<i>Id of labourer</i>

#### 15.Skill

<i>Column name</i>	<i>Data type</i>	<i>Constraints</i>	<i>Description</i>
<i>skill_id</i>	<i>Int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>category_id</i>	<i>Int</i>	<i>Foregin key</i>	<i>Id of skill</i>
<i>labourer_id</i>	<i>Int</i>	<i>Foregin key</i>	<i>Id of labourer</i>

#### 16.user

<i>Column name</i>	<i>Data type</i>	<i>constraints</i>	<i>Description</i>
<i>user_id</i>	<i>Int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>user_name</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Name of user</i>
<i>user_email</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>User email</i>
<i>user_phone_number</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Phone number</i>
<i>user_house_name</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>User house name</i>
<i>user_place</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>User place</i>
<i>user_post</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>User post</i>

#### 17.Payment

<i>Column name</i>	<i>Data type</i>	<i>Constraints</i>	<i>Description</i>
<i>payment_id</i>	<i>Int</i>	<i>Primary key</i>	<i>Unique identifier</i>
<i>bill_id</i>	<i>Int</i>	<i>Foergin key</i>	<i>Id of bill</i>
<i>payment_date</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Date of payment</i>
<i>payment_time</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Time of payment</i>
<i>acc_no</i>	<i>varchar(200)</i>	<i>Not null</i>	<i>Account number</i>
<i>amount</i>	<i>Int</i>	<i>Not null</i>	<i>Amount paid</i>

### **3.5. Data Flow Diagram**

A graphical representation is used to describe and analyses the movement of data through a system manual or automated including the processes, Storing of data and delays in the system. Data flow diagrams are the central tool and the basis from which other components are developed.

The transformation of data, from input to output through process may be described logically and independently of the physical components associated with the system.

They are termed logical dataflow diagrams, showing the actual implementations and the movement of data between people, departments and

workstations. DFD is one of the most important modelling tools used in system design. DFD shows the flow of data through different process in the system.

#### **PURPOSE:**

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have reasonably completed model of the behavior of the system. It is important to avoid premature designs, wherein develop designs before analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

Throughout my project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In my opinion, "efficient design of the data flow and context flow diagram helps to design the system successfully without much major flaws within the scheduled time". This is the most complicated part in a project. In the designing process, my project took more than the activities in the software lifecycle. If we design a system efficiently with all the future enhancements the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as a way of expressing system requirements in graphical form. A data flow diagram also known as "bubble chare" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It functionality



decomposes the requirement specification down to the lowest level. Data Flow Diagram depicts the

information flow, the transformation flow and the transformations that are applied as data move from input to output. Thus DFD describes what data flows rather than how they are processed.

Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is one of the most important tools used during system analysis. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

Data Flow Diagrams are made up of a number of symbols, which represents system components. Data flow modelling method uses four kinds of symbols, which are used to represent four kinds of system components.

These are

- Process
- Data stores
- Data flows
- External entity

**Process:**

Process shows the work of the system. Each process has one or more data inputs and produce one or more data outputs. Processes are represented by rounded rectangles in Data Flow Diagram. Each process has a unique name and number. This name and number appears inside the rectangle that represents the process in a Data Flow Diagram.

**Data Stores:**

A data stores is a repository of data. Processes can enter data, into a store or retrieve the data from the data store. Each data has a unique name.

**Data Flows:**

Data flows show the passage of data in the system and are represented by lines joining system components. An arrow indicates the direction of flow and the line is labelled by name of the dataflow.

**External Entity:**

External entities are outside the system but they either supply input data into the system or use other systems output. They are entities on which the designer has control. They may be an organizations customer or other bodies with which the system interacts. External entities that supply data into the system are sometimes called source. External entities that use the system data are sometimes called sinks. These are represented by rectangles in the

Data Flow Diagram.

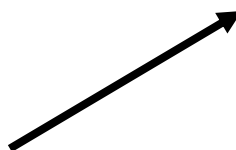
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Basic data flow diagram symbols are.....

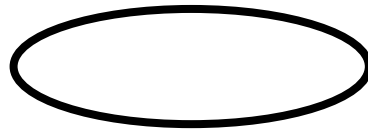
- A Square defines a source (originator) or destination of a system data:



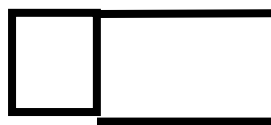
- An Arrow identifies data flow. It is a pipeline through which information flows:



- A Circle represents a process that transforms incoming data flow(s) into outgoing data flow(s):



- An Open Rectangle is a data store:

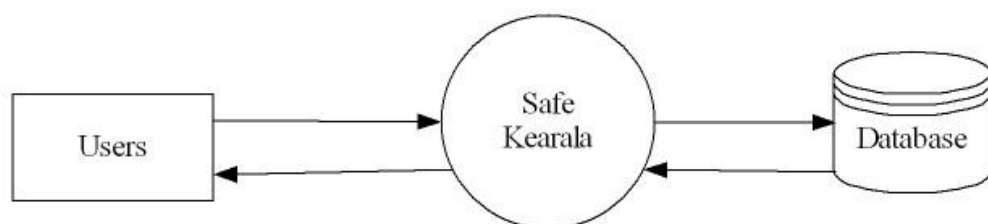


**Four steps are commonly used to construct a DFD:**

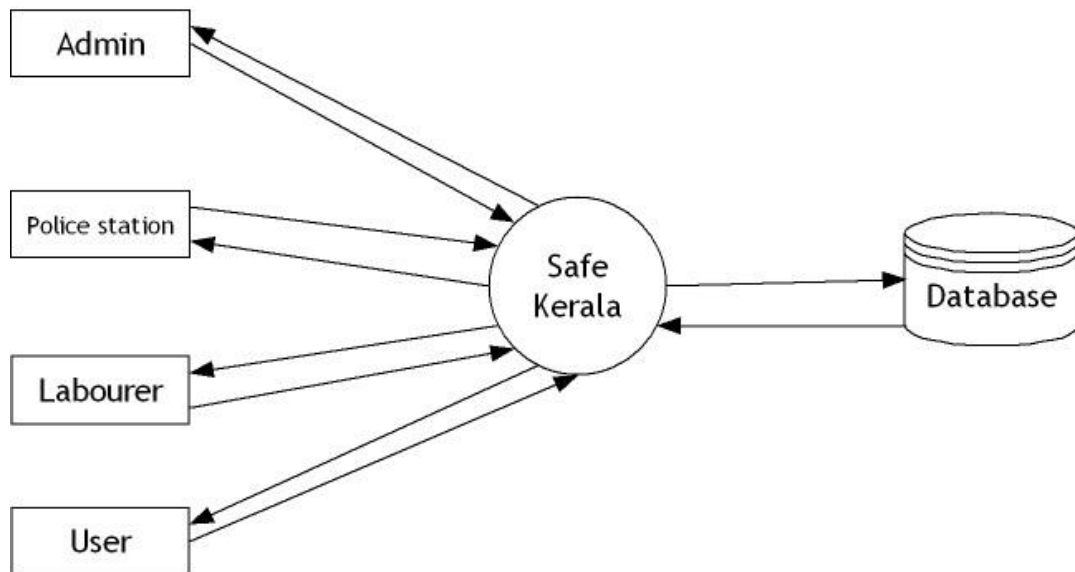
- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and left to right.
- When a process is exploded in to lower level details they are numbered.
- The names of data stores, sources and destinations are written in Capital letters.

*DFD Level-0*

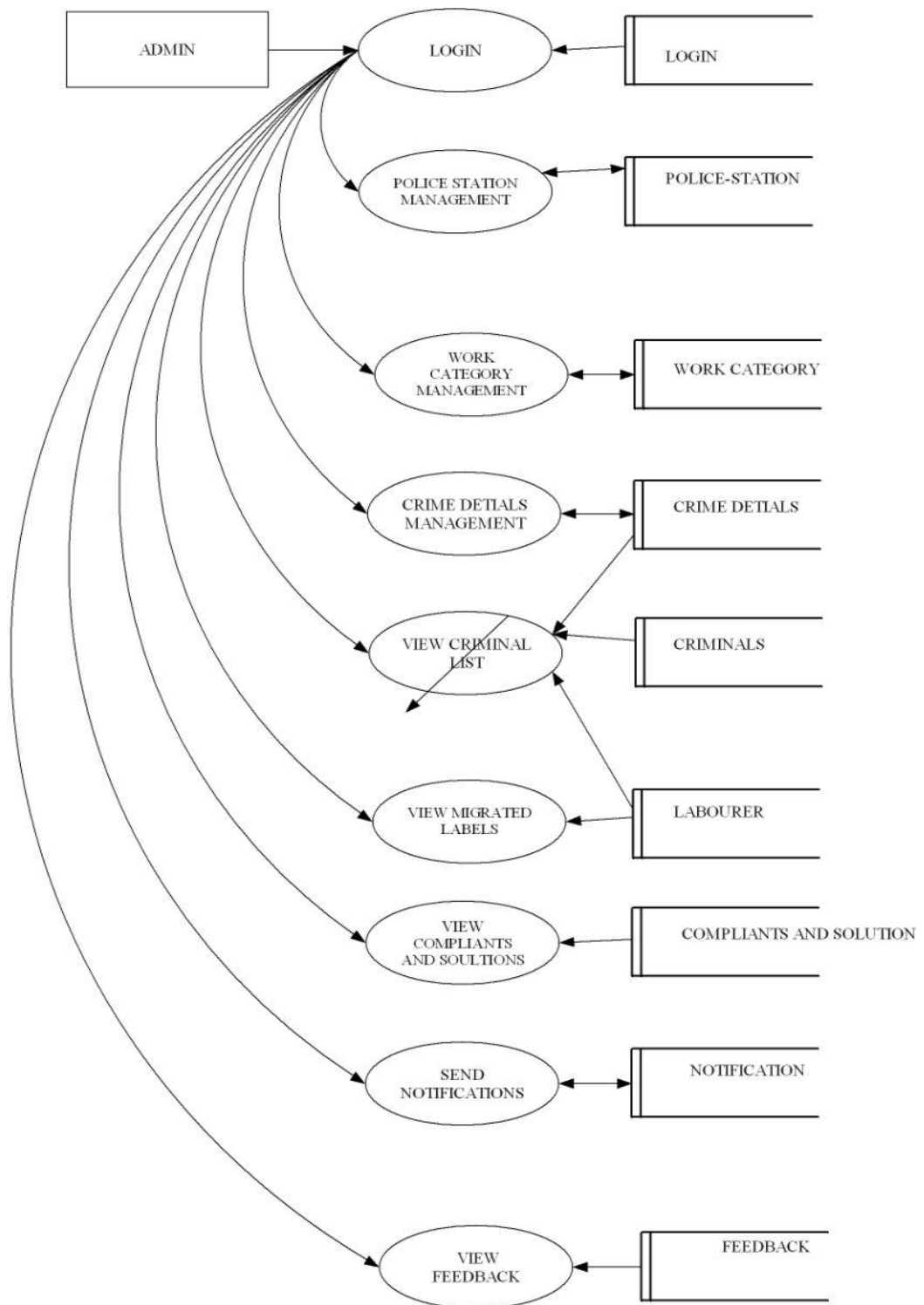
**Level 0**



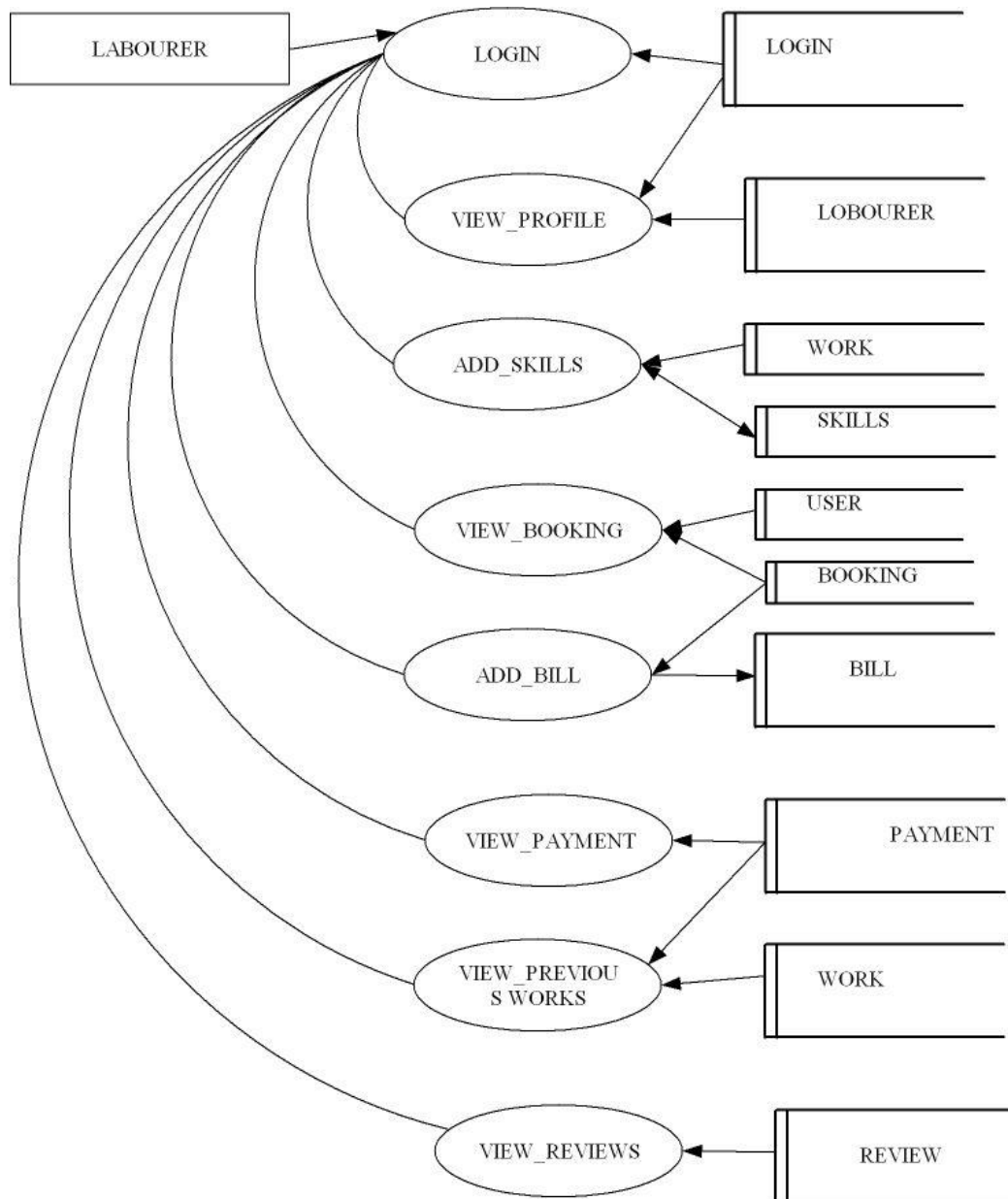
## Level 1



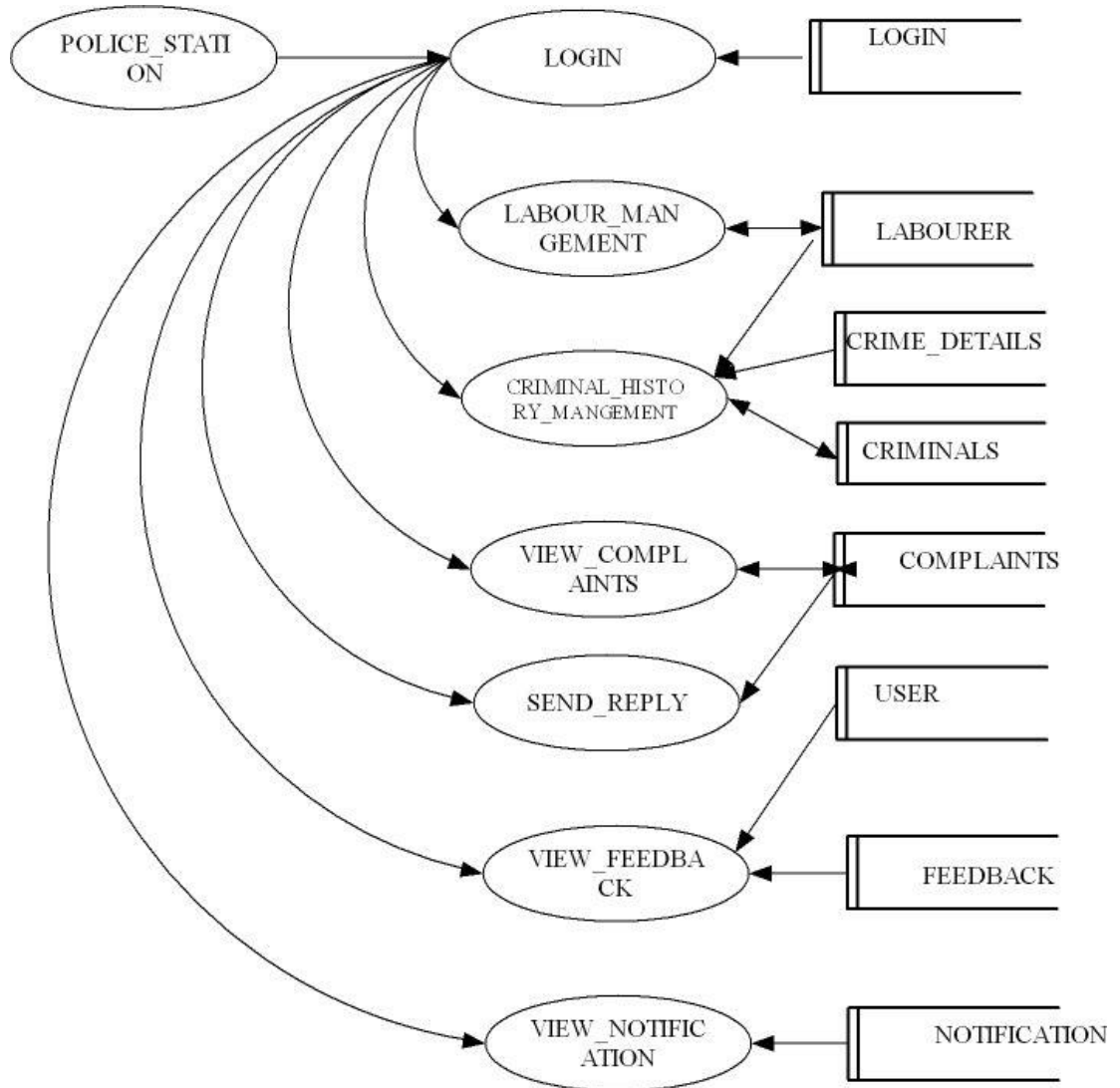
*DFD Level-1.1 Admin*



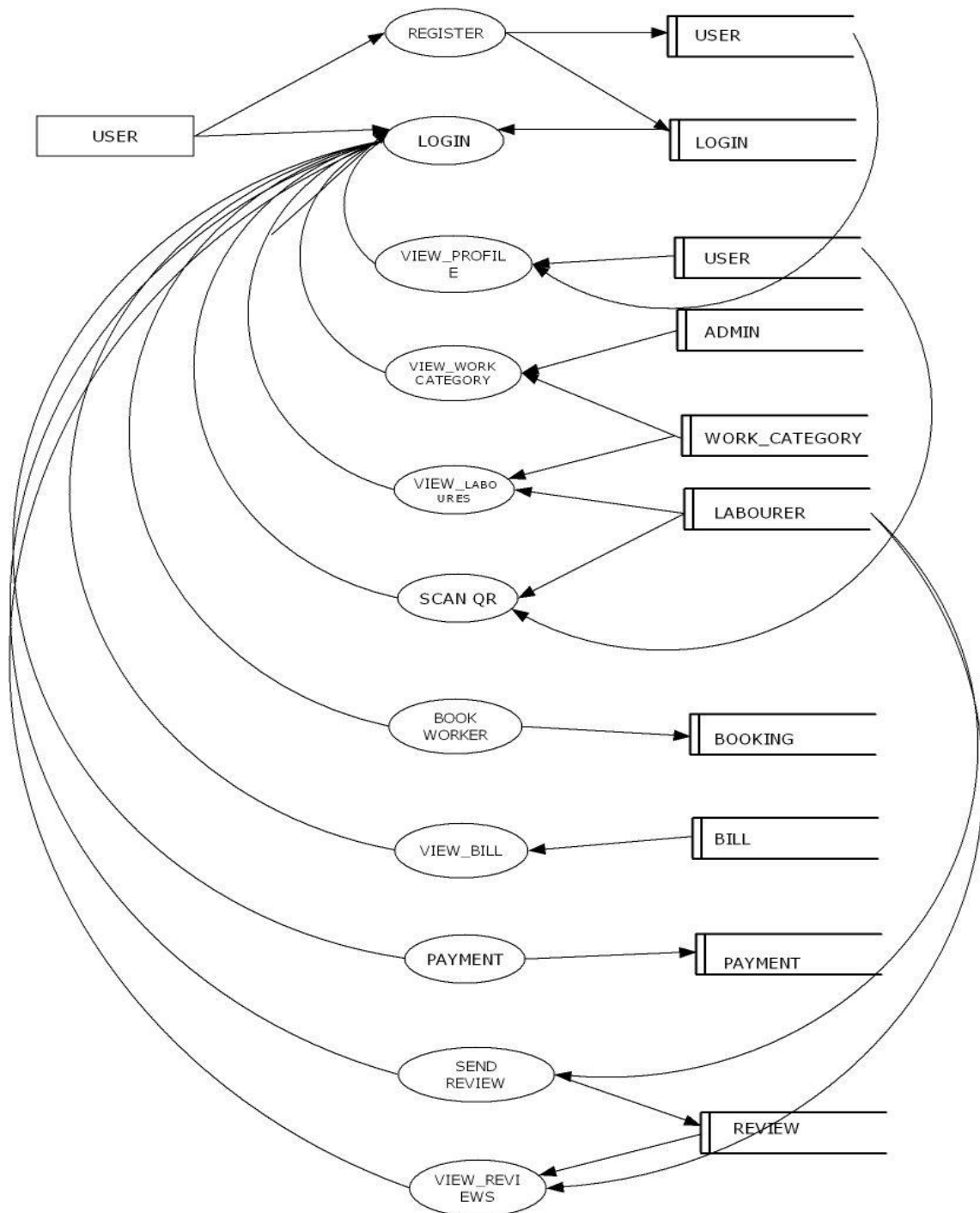
*DFD Level-1.2 Labourer*



*DFD Level-1.3 Police station*



*DFD Level-1.2 User*





### 3.6 ER Diagram

An ER diagram is a diagram that helps to design databases in an efficient way. It is a data model for describing the data or information. It is a visual representation of data that describes how data is related to each other. The main components of ER models are entities, attributes and the relationships that can exist among them.

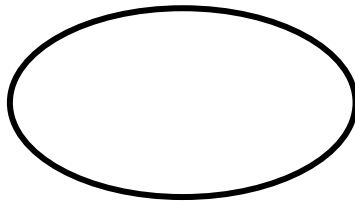
#### Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.



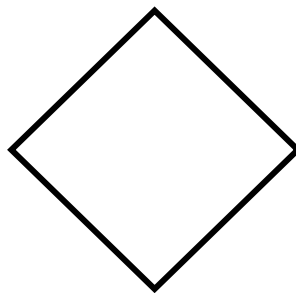
#### Attribute

Attributes are properties of entities. Attributes are represented by means of ellipses. Every ellipse represents one attribute and is directly connected to its entity (rectangle).



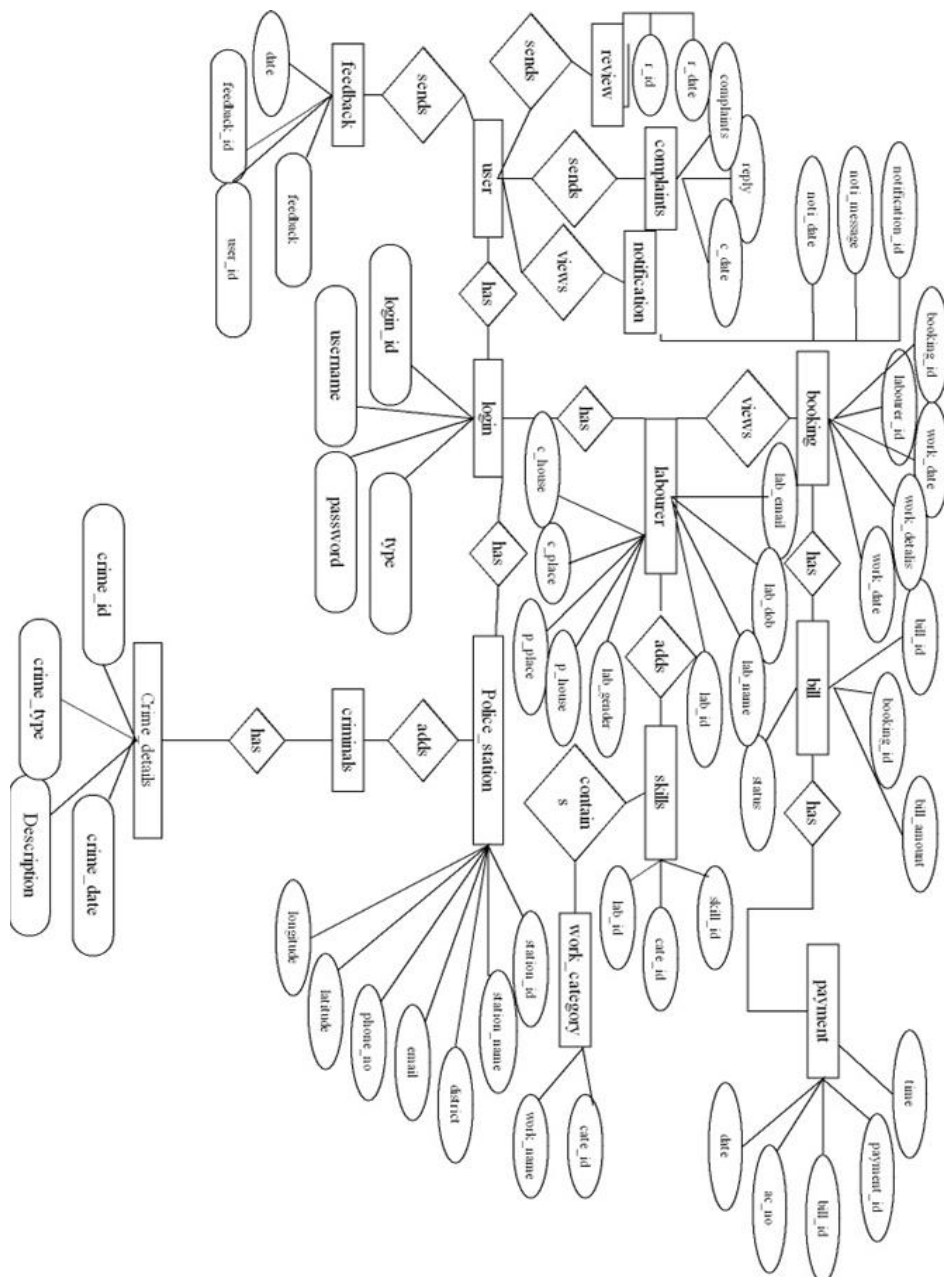
#### Relationship

Relationships are represented by diamond shaped box. Name of the relationship is written in the diamond box. All entities (rectangles), participating in relationship, are connected to it by a line.



*Architectural design*

# ENTITY RELATIONSHIP DIAGRAM



## **4.CODING**

## **4.1 INPUT INTERFACE**

Input design is a part of overall system design, which requires very careful attention. If data going into the system is correct, then the processing and output will magnify these errors. Thus the designer has a number of clear objectives in the different stages of input design.

- To produce a cost effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that input is acceptable to and understood by the user.

## **4.2 OUTPUT INTERFACE**

At the beginning of the output design various types of outputs such as external, internal, operational and interactive and turn around are defined. Then the format, content, location, frequency, volume and sequence of the outputs are specified. The content of the output must be defined in detail. The system analysis has two specific objectives at this stage.

- To interpret and communicate the results of the computer part of a system to the users in a form, which they can understand, and which meets their requirements.
- To communicate the output design specifications to programmers in a way in which it is unambiguous, comprehensive and capable of being translated into a programming language.

## **4.3 SOFTWARE DESCRIPTION**

### **4.3.1 HTML**

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML

specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

HTML files are written in ASCII text, so the user can use any text editor to create his/her web page, though a browser of one sort or another is necessary to view the web page. HTML is case insensitive with its language commands. The characters within the document, however, are case sensitive. The language consists of various "tags" which are known as elements. These allow the browser to understand (and put into the desired/specified format) the layout, background, headings, titles, lists, text and/or graphics on the page. The elements are classified according to their function in the HTML document. There are head elements and body elements. The head elements identify properties of the entire document, while body elements actually mark text as content and show a change in the appearance in one way or another. Most elements have a beginning and an ending which encompass the text the user wishes to mark with the tag. All HTML documents must begin with the element and end with the element. Some of the other elements which may be used are tags to create lists-- both ordered lists as well as unordered lists. The user may also create larger or smaller, bolder, italicized, or underlined text. Attributes may be used along with the elements. These perform functions such as placement of text, indication of the source files of images, and identification of links to the document or part of the document.

#### **4.3.2 CSS**

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages, user interfaces for web applications, and user interfaces for many mobile applications. CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colours, and fonts.

## **Advantages of CSS**

- CSS saves time – you can write CSS once and then reuse same sheet in multiple HTML pages.  
You can define a style for each HTML element and apply it to as many Web pages as you want.
- Pages load faster – If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
- Easy maintenance – To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- Superior styles to HTML – CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- Multiple Device Compatibility – Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- Global web standards – Now HTML attributes are being deprecated and it is being recommended to use CSS. So it is a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.
- Offline Browsing – CSS can store web applications locally with the help of an offline cache. Using of this, we can view offline websites. The cache also ensures faster loading and better overall performance of the website.
- Platform Independence – The Script offer consistent platform independence and can support latest browsers as well

### **4.4.3 JAVASCRIPT**

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript.

The General-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

Advantages of JavaScript:

- Less server interaction – you can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.
- Immediate feedback to the visitors – They don't have to wait for a page reload to see if they have forgotten to enter something.
- Increased interactivity – you can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.
- Richer interfaces – you can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

#### **4.3.4 MySQL**

MySQL is an open-source relational database management system (RDBMS). MySQL is released under an open-source license. So you have nothing to pay to use it. MySQL is a very powerful program in its own right.

It handles a large subset of the functionality of the most expensive and powerful database packages. It uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.

It works very quickly and works well even with large data sets. It is very friendly to PHP, the most appreciated language for web development. It supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB). It is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

## Major features as available in MySQL 5.6

- A broad subset of ANSI SQL 99, as well as extensions.
- Cross-platform support.
- Stored procedures, using a procedural language that closely adheres to SQL/PSM.
- Triggers.
- Cursors.
- Updatable views.
- Online DDL when using the InnoDB Storage Engine.
- Information schema.
- Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
- A set of SQL Mode options to control runtime behaviour, including a strict mode to better adhere to SQL standards.
- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine.
- Transactions with savepoints when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
- ACID compliance when using InnoDB and NDB Cluster Storage Engines.
- SSL support → Query caching → Sub-SELECTs (i.e. nested SELECTs) .
- Built-in replication support (i.e., master-master replication and master-slave replication) with one master per slave, many slaves per master.
- Multi-master replication is provided in MySQL Cluster, and multimaster support can be added to unclustered configurations using Galera Cluster.
- Full-text indexing and searching.
- Embedded database library.
- Unicode support.
- Partitioned tables with pruning of partitions in optimizer.
- Shared-nothing clustering through MySQL Cluster.
- Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.
- Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.



- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

## **Advantages**

MySQL database server has lots of advantages over its competitors. Some of these advantages have been explained below.

- Open Source and Cost Effective:

The best thing about MySQL server is that this is open source and it has a free version as well.

By open source software, we mean that the code of the software is available and anyone can tailor it according to his requirement. Companies prefer MySQL because they don't have to pay anything for this excellent product.

- Portability:

MySQL is cross platform database server. MySQL can be run on a variety of platforms including Windows, OS2, Linux and Solaris. Portability of MySQL server makes it suitable for applications that target multiple platforms particularly web application. MySQL contains API for almost all the major programming languages and can be easily integrated with the languages like PHP, C++, Perl, C, Python and ruby. In fact, MySQL is a part of the famous LAMP (Linux Apache MySQL PHP) server stack which is used worldwide for web application development.

- Seamless Connectivity:

Various secure and seamless connection mechanisms are available in order to connect with MySQL server. These connections include named pipes, TCP/IP sockets and UNIX Sockets.

- Rapid Development and Continuous Updates:

Being an open source product, MySQL has a very large developer community which releases regular patches and updates for MySQL. Several database templates have been developed which can be readily used and modified resulting in rapid application development.

- Security:

MySQL server databases are extremely secure and all the data access scenarios are protected via password and good thing about these passwords is that they are stored in encrypted form and it is not easy to break these advanced and complex encryption algorithms.

#### **4.3.5 Python**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

#### **Major features of python**

- Easy to code
- Free and Open Source
- Object-Oriented Language
- GUI Programming Support

- High-Level Language
- Extensible feature
- Python is **Portable** language
- Python is Integrated language

#### **Advantages**

- Extensive support libraries
- Integration feature
- Improved productivity
- Easy to learn and write
- Vast library support
- Free and open source

#### **4.3.6 PyCharm**

**PyCharm** is an Integrated Development Environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains (formerly known as IntelliJ). It provides code analysis, a graphical debugger, an integrated unit tester, integration with Version Control Systems (VCSes), and supports web development with Django as well as data science with Anaconda

#### **FEATURES**

- Coding assistance and analysis, with code completion, syntax and error highlighting, linear integration, and quick fixes
- Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages

Python refactoring: includes rename, extract method, introduce variable, introduce constant, pull up, push down and others

- Integrated Python debugger
- Integrated unit testing, with line-by-line code coverage

Version control integration: unified user interface

for Mercurial, Git, Subversion, Perforce and CVS with change lists and merge

#### **4.3.6 Flask**

Flask is a web application framework written in Python. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Pocco. Flask is based on the Werkzeug WSGI toolkit and the Jinja2 template engine. Both are Pocco projects.

It is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier. It gives developers flexibility and is a more accessible framework for new developers since you can build a web application quickly using only a single Python file. Flask is also extensible and doesn't force a particular directory structure or require complicated boilerplate code before getting started.

### **Android**

**Android** is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language. Android applications are written in the Java programming language. The Android SDK tools compile the code—along with any data and resource files—into an Android package, an archive file with an .apk suffix. All the code in a single .apk file is considered to be one application and is the file that Android-powered devices use to install the application. Application components are the essential building blocks of an Android application. Each component is a different point through which the system can enter your application. Not all components are actual entry points for the user and some depend on each other, but each one exists as its own entity and plays a specific role—each one is a unique building block that helps define application's overall behaviour.

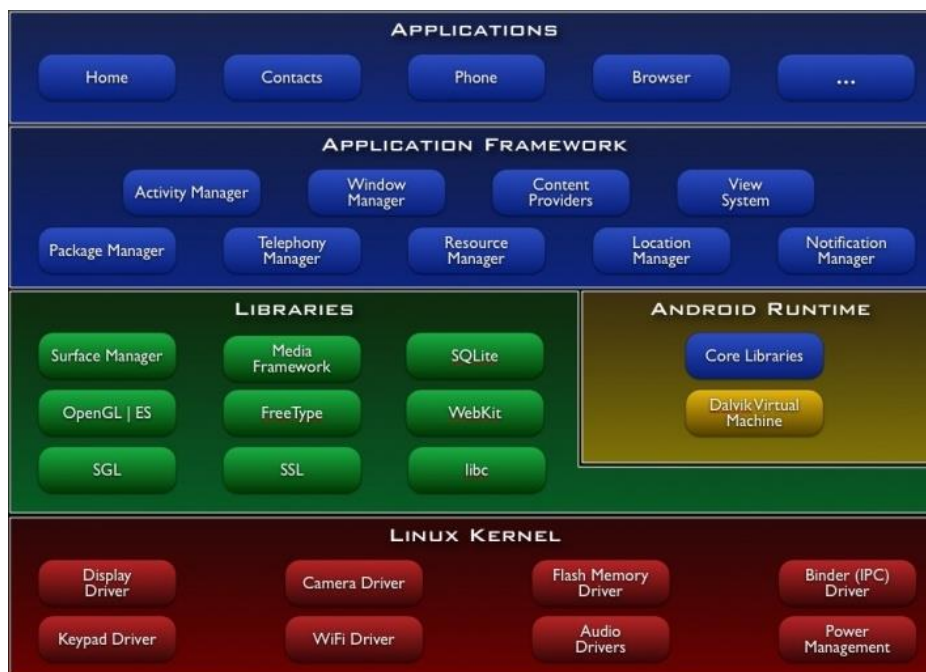
#### **Features**

- Application framework enabling reuse and replacement of components
- Dalvik virtual machine optimized for mobile devices
- Integrated browser based on the open source Web Kit engine

- Optimized graphics powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
- Media support for common audio, video, and still image formats (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
- GSM Telephony (hardware dependent)
- Bluetooth, EDGE, 3G, and Wi-Fi (hardware dependent)
- Camera, GPS, compass, and accelerometer (hardware dependent)
- Rich development environment including a device emulator, tools for debugging, memory and performance profiling, and a plug-in for the Eclipse IDE.

## ANDROID ARCHITECTURE

The following diagram shows the major components of the Android operating system. Each section is described in more detail below.



## APPLICATION FRAMEWORK

By providing an open development platform, Android offers developers the ability to build extremely rich and innovative applications. Developers are free to take advantage of the device hardware, access location information, run background services, set alarms, add notifications to the status bar, and much, much more.

Developers have full access to the same framework APIs used by the core applications. The application architecture is designed to simplify the reuse of components; any application can publish its capabilities and any other application may then make use of those capabilities (subject to security constraints enforced by the framework). This same mechanism allows components to be replaced by the user.

Underlying all applications is a set of services and systems, including:

A rich and extensible set of the views that can be used to build an application, including lists, grids, text boxes, buttons, and even an embeddable web browser

Content Providers that enable applications to access data from other applications (such as Contacts), or to share their own data

A Resource Manager, providing access to non-code resources such as localized strings, graphics, and layout files

A Notification Manager that enables all applications to display custom alerts in the status bar

An Activity Manager that manages the lifecycle of applications and provides a common navigation back stack.

## **Libraries**

Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below:

- System C library - a BSD-derived implementation of the standard C system library (libc), tuned for embedded Linux-based devices
- Media Libraries - based on Packet Video's Open CORE; the libraries support playback and recording of many popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG
- Surface Manager - manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications

- LibWebCore - a modern web browser engine which powers both the Android browser and an embeddable web view
- SGL - the underlying 2D graphics engine
- 3D libraries - an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer
- Free Type - bitmap and vector font rendering<sup>24</sup>

## **Android Runtime**

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language. Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple VMs efficiently. The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been transformed into the .dex format by the included "dx" tool. The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management. Linux Kernel Android relies on Linux version 2.6 for core system services such as security, memory management, process management, network stack, and driver model. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack.

## **Activity Lifecycle**

Activities in the system are managed as an activity stack. When a new activity is started, it is placed on the top of the stack and becomes the running activity -- the previous activity always remains below it in the stack, and will not come to the foreground again until the new activity exits.

An activity has essentially four states:

- If an activity is in the foreground of the screen (at the top of the stack), it is active or running.
- If an activity has lost focus but is still visible (that is, a new non-full sized or transparent activity has focus on top of your activity), it is paused. A

paused activity is completely alive (it maintains all state and member information and remains attached to the window manager), but can be killed by the system in extreme low memory situations.

- If an activity is completely obscured by another activity, it is stopped. It still retains all state and member information, however, it is no longer visible to the user so its window is hidden and it will often be killed by the system when memory is needed elsewhere.
- If an activity is paused or stopped, the system can drop the activity from memory by either asking it to finish, or simply killing its process. When it is displayed again to the user, it must be completely restarted and restored to its previous state.

### 4.3.8 Android Studio

**Android Studio** is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (ADT) as primary IDE for native Android application development. Android Studio supports all the same programming languages of IntelliJ, and PyCharm and Android Studio 3.0 supports Java 7 language features and a subset of Java 8 language features that vary by platform version. Features like Gradle-based build support, Android-specific refactoring and quick fixes, a rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations, Android Virtual Device (Emulator) to run and debug apps in the Android studio, etc. are provided in the current stable version.



## **5.CODING PAGES**

## 5.1 Admin page

```
import smtplib
from email.mime.text import MIMEText

from flask import Flask,render_template,request,redirect,session,jsonify
import random
import datetime
from DBConnection import Db
app = Flask(__name__)
app.secret_key="safekerala"
@app.route('/',methods={'get','post'})
def login():
    db=Db()
    if request.method == "POST":
        Username=request.form['textfield']
        password=request.form['textfield2']
        qry=db.selectOne("select * from login where username='"+Username+"' and
password='"+password+"'")
        # res=db.insert("insert in to login values(
null,username='"+Username+"',password='"+password+"'")
        if qry is not None:
            if qry['usertype']=='admin':
                session['lg']='lin'
                return redirect('/homepage')
            elif qry['usertype']=='police_station':
                session['lid']=qry['login_id']
                session['lg']='lin'

                return redirect('/police_homepage')
            elif qry['usertype']=='labour':
                session['lid']=qry['login_id']
                session['lg']='lin'
                return redirect('/labourer_homepage')
            else:
                return '<script>alert("invalid user");window.location="/"</script>'
        else:
            return '<script>alert("invalid password");window.location="/"</script>'
        else:
            return render_template('login_index.html')
@app.route('/police_station')
def police_station():
    if session['lg']!='lin':
        return redirect('/')
    return render_template('admin/police_station.html')
@app.route('/police_station_post',methods=['post'])
def police_station_post():
```

```

if session['lg']!= 'lin':
    return redirect('/')
station_name=request.form['textfield']
district=request.form['select']
email=request.form['textfield13']
phone_no=request.form['textfield4']
latitude=request.form['textfield2']
Longitude=request.form['textfield3']
password=random.randint(000,9999)
db=Db()
qry1=db.selectOne("select * from login where username='"+email+"'")
if qry1 is not None:
    return "<script>alert('Email Already added');
window.location='/police_station';</script>"

qry=db.insert("insert into login VALUES
('"+email+"','"+str(password)+"','police_station')")
db.insert("insert into police_station VALUES
('"+str(qry)+"','"+station_name+"','"+district+"','"+email+"','"+phone_no+"','"+latitude
+"','"+Longitude+"')")
try:
    gmail = smtplib.SMTP('smtp.gmail.com', 587)
    gmail.ehlo()
    gmail.starttls()
    gmail.login('safekearala@gmail.com', 'wgyituaipitiikivk')
except Exception as e:
    print("Couldn't setup email!!" + str(e))
msg = MIMEText("Your Password is " + str(password))
msg['Subject'] = 'Safe kerala Website!!'
msg['To'] = email
msg['From'] = 'compnaymail@gmail.com'
try:
    gmail.send_message(msg)
except Exception as e:
    print("COULDN'T SEND EMAIL", str(e))
return "<script>alert('police station added');
window.location='/police_station';</script>"
@app.route('/view_policestation')
def view_policestation():
    if session['lg']!= 'lin':
        return redirect('/')
    db=Db()
    qry=db.select("select * from police_station")
    return render_template('admin/view_policestation.html',data=qry)
@app.route('/edit_policestation_1/<pid>')
def edit_policestation_1(pid):

```

```

    if session['lg'] != 'lin':
        return redirect('/')
    db = Db()
    res = db.selectOne("select * from police_station where station_id='" + pid + "'")
    return render_template('admin/edit_policestation.html', data=res, station_id=pid)
@app.route('/edit_policestation', methods={'post'})
def edit_policestation():
    if session['lg'] != 'lin':
        return redirect('/')
    policeid = request.form['pid']
    station_name = request.form['textfield']
    district = request.form['select']
    email = request.form['textfield12']
    phone_no = request.form['textfield13']
    latitude = request.form['textfield2']
    Longitude = request.form['textfield3']
    db = Db()
    db.update("update police_station set
station_name='" + station_name + "',district='" + district + "',email='" + email + "',phone='" +
phone_no + "',latitude='" + latitude + "',longitude='" + Longitude + "' where
station_id='" + policeid + "'")
    return '<script>alert("updated
successfully");window.location="/view_policestation"</script>'
@app.route('/delete_station/<sid>')
def delete_station(sid):
    if session['lg'] != 'lin':
        return redirect('/')
    db = Db()
    qry = db.delete("delete from police_station WHERE station_id='" + sid + "'")
    qry = db.delete("delete from login WHERE login_id='" + sid + "'")
    return redirect('/view_policestation')
@app.route('/add_workcategory')
def add_workcategory():
    if session['lg'] != 'lin':
        return redirect('/')
    return render_template('admin/add_workcategory.html')
@app.route('/add_workcategory_post', methods=['post'])
def add_workcategory_post():
    if session['lg'] != 'lin':
        return redirect('/')
    Work_name = request.form['textarea']
    db = Db()
    qry = db.selectOne("select * from work_category where work_name='" +
Work_name + "'")
    if qry is not None:
        return "<script>alert('Already added');

```

```

window.location='/add_workcategory';</script>"
    db.insert("insert into work_category VALUES('"+Work_name+"') ")
    return "<script>alert('Work category added');
window.location='/add_workcategory';</script>"
@app.route('/view_workcategory')
def view_workcategory():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    qry=db.select("select * from work_category")
    return render_template('admin/view_workcategory.html',data=qry)
@app.route('/edit_workcategory/<cid>')
def edit_workcategory(cid):
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    res=db.selectOne("select * from work_category WHERE category_id='"+cid+"'")
    print(res)
    return render_template('admin/edit_workcategory.html',data=res,category_id=cid)
@app.route('/edit_workcategory_post',methods=['post'])
def edit_workcategory_post():
    if session['lg']!='lin':
        return redirect('/')
    cid=request.form['cid']
    workname=request.form['textfield']
    db=Db()
    db.update("update work_category set work_name='"+workname+"' where
category_id='"+cid+"' ")
    return "'<script>alert('update');window.location='/view_workcategory'</script>'"
@app.route('/delete_workcategory/<cid>')
def delete_workcategory(cid):
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    qry=db.delete("delete from work_category where category_id='"+cid+"'")
    return '<script>alert("delete
successfully");window.location="/view_workcategory"</script>'
@app.route('/view_crimedetails')
def view_crimedetails():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    qry = db.select("select * from crime_details")
    return render_template('admin/view_crimedetails.html',data=qry)
@app.route('/delete_crimedetails/<cid>')
def delete_crimedetails(cid):

```

```

if session['lg']!= 'lin':
    return redirect('/')
db=Db()
qry = db.delete("delete from crime_details WHERE crime_id='"+cid+"'")
return redirect('/view_crimedetails')
@app.route('/add_crimedetails')
def add_crimedetails():
    if session['lg']!= 'lin':
        return redirect('/')
    return render_template('admin/add_crimedetails.html')
@app.route('/add_crimedetails_post',methods=['post'])
def add_crimedetails_post():
    if session['lg']!= 'lin':
        return redirect('/')
    Crime_type=request.form['textarea']
    Crime_description=request.form['textarea2']
    db=Db()
    qry=db.selectOne("select * from crime_details where
crime_type='"+Crime_type+"' and crime_description='"+Crime_description+"'")
    if qry is not None:
        return "<script>alert('Already added');
window.location='/add_crimedetails';</script>"
    db.insert("insert into crime_details VALUES
('"+Crime_type+"','"+Crime_description+"', curdate())")
    return "<script>alert('Crime details added');
window.location='/add_crimedetails';</script>"
@app.route('/edit_crime_details/<cid>',methods={'get','post'})
def edit_crimedetails(cid):
    if session['lg']!= 'lin':
        return redirect('/')
    if request.method=="POST":
        Crime_type = request.form['textfield1']
        Crime_description = request.form['textarea2']
        # crime_date=request.form['textfield']
        db=Db()
        db.update("update crime_details set crime_type='"+
Crime_type+"',crime_description='"+Crime_description+"' where
crime_id='"+cid+"'")
        return "<script>alert('Crime details edited');
window.location='/view_crimedetails';</script>"
    else:
        db=Db()
        qry=db.selectOne("select * from crime_details where crime_id='"+cid+"'")
        return render_template('admin/edit_crimedetails.html',data=qry)

```

```

@app.route('/view_criminals')
def view_criminals():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    qry = db.select("select * from crime_details,criminals,police_station,labourer
where criminals.station_id=police_station.station_id and
criminals.labour_id=labourer.labourer_id and
criminals.crime_id=crime_details.crime_id ")
    return render_template('admin/view_criminals.html',data=qry)
@app.route('/view_labourer')
def view_labourer():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    qry = db.select("select * from labourer")
    return render_template('admin/view_labourer.html',data=qry)
@app.route('/view_complanits')
def view_complanits():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    qry = db.select("select * from complaints,user where
complaints.user_id=user.user_id")
    return render_template('admin/view_complanits.html',data=qry)
@app.route('/send_notification')
def send_notification():
    if session['lg']!='lin':
        return redirect('/')
    return render_template('admin/send_notification.html')
@app.route('/send_notification_post',methods=['post'])
def send_notification_post():
    if session['lg']!='lin':
        return redirect('/')
    Notification_message=request.form['textarea']
    # Date=request.form['textfield']
    db=Db()
    qry=db.selectOne("select * from notification where
notification_message='"+Notification_message+"'")
    if qry is not None:
        return "<script>alert('Already added');
window.location='/send_notification';</script>"
    db.insert("insert into notification values ('','"+Notification_message+"',curdate())")
    return "<script>alert('Notification sent'); window.location='/homepage';</script>"

```

```

@app.route('/delete_notification/<nid>')
def delete_notification(nid):
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    qry=db.delete("delete from notification WHERE notification_id='"+nid+"'")
    return redirect('/admin_view_notification#services')
@app.route('/admin_view_notification')
def admin_view_notification():
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    qry = db.select("select * from notification")
    return render_template('admin/view_notification.html',data=qry)
@app.route('/view_feedback')
def view_feedback():
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    qry = db.select("select * from feedback,user where feedback.user_id=user.user_id")
    return render_template('admin/view_feedback.html',data=qry)
@app.route('/homepage')
def homepage():
    if session['lg'] != 'lin':
        return redirect('/')
    return render_template('admin/admin_index.html')
@app.route('/send_reply/<did>')
def send_reply(did):
    if session['lg'] != 'lin':
        return redirect('/')
    return render_template('admin/send_reply.html',did=did)
@app.route('/send_reply_post',methods=['post'])
def send_reply_post():
    if session['lg'] != 'lin':
        return redirect('/')
    # if request.method=="POST":
    db=Db()
    hid=request.form['did']
    reply=request.form['textarea']
    db.update("update complaints set reply='"+reply+"'where
complaint_id='"+hid+"'")
    return '<script>alert("reply sended
successfully");window.location="/view_complanits"</script>'
    # else:
    #     return render_template("admin/send_reply.html")

```



```

#
=====
=====
=====
#                                POLICE STATION
#
=====
=====
=====
@app.route('/police_homepage')
def police_homepage():
    if session['lg']!= 'lin':
        return redirect('/')
    return render_template('police_station/police_index.html')
@app.route('/add_criminal/<lbid>')
def add_criminal(lbid):
    if session['lg']!= 'lin':
        return redirect('/')
    db=Db()
    qry=db.select("select * from crime_details")
    return render_template('police_station/add_criminal.html',lbid=lbid,data=qry)
@app.route('/add_criminal_post',methods=['post'])
def add_criminal_post():
    if session['lg']!= 'lin':
        return redirect('/')
    crime_type=request.form['ct']
    crime_date=request.form['d']
    lbid=request.form['lb']
    db=Db()
    db.insert("insert into criminals VALUES
('"+str(session['lid'])+"','"+str(lbid)+"','"+crime_type+"','"+crime_date+"',curdate())"
)
    return '<script>alert("added
successfully");window.location="/view_labourer_service"</script>'

@app.route('/add_labour')
def add_labour():
    if session['lg']!= 'lin':
        return redirect('/')
    return render_template('police_station/add_labour.html')
@app.route('/add_labour_post',methods=['post'])
def add_labour_post():
    if session['lg']!= 'lin':
        return redirect('/')
    name=request.form['textfield']
    dob=request.form['textfield2']

```

```

photo = request.files['fileField']
Details = request.form['textarea']
Phone_no = request.form['textfield3']
email = request.form['textfield4']
Gender= request.form['RadioGroup1']
Proper_place= request.form['textfield5']
Proper_house= request.form['textfield6']
Proper_post= request.form['textfield7']
Current_place=request.form['textfield8']
Current_house=request.form['textfield10']
Current_post=request.form['textfield11']
adhr_no=request.form['textfield12']
password=random.randint(0000,9999)
date=datetime.datetime.now().strftime("%Y%m%d-%H%M%S")
photo.save(r"C:\Users\user\PycharmProjects\safekerala\static\pic\\"+date+'.jpg')
path="/static/pic/"+date+'.jpg'
db=Db()
qry1 = db.selectOne("select * from login where username='" + email + "'")
if qry1 is not None:
    return "<script>alert('Email Already added');
window.location=/add_labour';</script>"
    qry=db.insert("insert into login VALUES
('"+email+"','"+str(password)+"','labour')")
    q=db.insert("insert into labourer
(labourer_id,labourer_name,labourer_dob,labourer_details,labourer_photo,labourer_p
hone_number,labourer_email,labourer_gender,proper_place,proper_house,proper_pos
t,current_place,current_house,current_post,Aadhar_no) VALUES
('"+str(qry1)+"','"+name+"','"+dob+"','"+Details+"','"+str(path)+"','"+Phone_no+"','"+e
mail+"','"+Gender+"','"+Proper_place+"','"+Proper_house+"','"+Proper_post+"','"+Cur
rent_place+"','"+Current_house+"','"+Current_post+"','"+adhr_no+"')")
import qrcode
# Create qr code instance
qr = qrcode.QRCode(
    version=1,
    error_correction=qrcode.constants.ERROR_CORRECT_H,
    box_size=3,
    border=4,
)
# The data that you want to store
# data = "The Data that you need to store in the QR Code"
data = q
# Add data
qr.add_data(data)
qr.make(fit=True)
# Create an image from the QR Code instance
img = qr.make_image()

```

```

img.save(r"C:\Users\user\PycharmProjects\safekerala\static\qrcores\\"+str(data)+'.jpg'
)
try:
    gmail = smtplib.SMTP('smtp.gmail.com', 587)
    gmail.ehlo()
    gmail.starttls()
    gmail.login('safekearala@gmail.com', 'wgyituaipitiikivk')
except Exception as e:
    print("Couldn't setup email!!" + str(e))
msg = MIMEText("Your Password is " + str(password))
msg['Subject'] = 'Safe kerala Website!!'
msg['To'] = email
msg['From'] = 'compnaymail@gmail.com'
try:
    gmail.send_message(msg)
except Exception as e:
    print("COULDN'T SEND EMAIL", str(e))
return '<script>alert("Added
successfully");window.location="/police_homepage"</script>'

@app.route('/send_reply_service/<did>')
def send_reply_service(did):
    if session['lg'] != 'lin':
        return redirect('/')
    return render_template('admin/send_reply.html',did=did)

@app.route('/send_reply_post_sevice',methods=['post'])
def send_reply_post_sevice():
    if session['lg'] != 'lin':
        return redirect('/')
    # if request.method=="POST":
    db=Db()
    hid=request.form['did']
    reply=request.form['textarea']
    db.update("update complaints set reply='"+reply+"'where
complaint_id='"+hid+"'")
    return '<script>alert("reply sended
successfully");window.location="/view_complanits"</script>'
    # else:
    #     return render_template("admin/send_reply.html")
@app.route('/view_notification')
def view_notification():
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    qry=db.select("select * from notification")

```

```

    return render_template('police_station/view_notification.html',data=qry)
@app.route('/view_criminals_station/<labour_id>')
def view_criminals_station(labour_id):
    if session['lg']!= 'lin':
        return redirect('/')
    db=Db()
    qry=db.select("select * from criminals,crime_details where
criminals.labour_id='"+labour_id+"' and criminals.crime_id = crime_details.crime_id
")
    return render_template('police_station/view_criminals.html',data=qry)
@app.route('/view_labourer_service')
def view_labourer_service():
    if session['lg']!= 'lin':
        return redirect('/')
    db=Db()
    qry=db.select("select * from labourer ")
    print(qry)
    return render_template('police_station/view_labourer.html',data=qry)
@app.route('/view_user_complaints_service')
def view_user_complaints_service():
    if session['lg']!= 'lin':
        return redirect('/')
    db=Db()
    qry = db.select("select * from complaints,user where
complaints.user_id=user.user_id and station_id='"+str(session['lid'])+"'")
    return render_template('police_station/view_user_complaints.html',data=qry)
@app.route('/view_feedback_service')
def view_feedback_service():
    if session['lg']!= 'lin':
        return redirect('/')
    db=Db()
    qry = db.select("select * from feedback,user where feedback.user_id=user.user_id")
    return render_template('police_station/view_feedback.html',data=qry)
@app.route('/delete_criminals/<cid>')
def delete_criminals(cid):
    if session['lg']!= 'lin':
        return redirect('/')
    db=Db()
    qry=db.delete("delete from labourer WHERE labourer_id='"+cid+"'")
    qry=db.delete("delete from login WHERE login_id='"+cid+"'")
    return '<script>alert("delete
successfully");window.location="/view_labourer_service"</script>'
@app.route('/delete_lb/<cid>')
def delete_lb(cid):
    if session['lg']!= 'lin':
        return redirect('/')

```

```

db=Db()
qry=db.delete("delete from criminals WHERE criminal_id='"+cid+"'")
return '<script>alert("delete
successfully");window.location="/view_labourer_service"</script>'
##### LABOURER
#####
#####
#####
#####

@app.route('/labourer_homepage')
def labourer_homepage():
    if session['lg']!= 'lin':
        return redirect('/')
    return render_template("labourer/labour_index.html")
@app.route('/labourer_profile')
def labourer_profile():
    if session['lg']!= 'lin':
        return redirect('/')
    db=Db()
    res=db.selectOne("select * from labourer where
labourer_id='"+str(session['lid'])+"'")
    return render_template("labourer/view_profile.html", data=res)

@app.route('/view_workcategory_labour')
def view_workcategory_labour():
    if session['lg']!= 'lin':
        return redirect('/')
    db=Db()
    qry=db.select("select * from work_category")
    return render_template('labourer/view_work.html',data=qry)
@app.route('/Add_skill/<cid>')
def Add_skill(cid):
    if session['lg']!= 'lin':
        return redirect('/')
    db=Db()
    qry=db.selectOne("select * from skill where category_id='"+cid+"' and
labourer_id='"+str(session['lid'])+"'")
    if qry is not None:
        return '<script>alert("already added
");window.location="/view_workcategory_labour"</script>'
    else:
        db.insert("insert into skill VALUES ('','"+cid+"','"+str(session['lid'])+"'")
        return '<script>alert("added
successfully");window.location="/view_workcategory_labour"</script>'

```

```

@app.route('/view_skill')
def view_skill():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    qry = db.select("select * from work_category,skill WHERE
skill.category_id=work_category.category_id")
    return render_template('labourer/View_skill.html',data=qry)

@app.route('/delete_skill/<sid>')
def delete_skill(sid):
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    qry=db.delete("delete from skill WHERE skill_id='"+sid+"'")
    return '<script>alert("delete successfully");window.location="/view_skill"</script>'

@app.route('/view_booking')
def view_booking ():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    qry=db.select("select * from booking,user,skill,work_category WHERE
booking.user_id=user.user_id AND booking.skill_id=skill.skill_id AND
skill.category_id=work_category.category_id")
    return render_template('labourer/View_booking.html',data=qry)

@app.route('/view_previous_work')
def view_previous_work ():
    if session['lg']!='lin':
        return redirect('/')
    db=Db()
    qry=db.select("select * from booking,user,skill,work_category WHERE
booking.user_id=user.user_id AND booking.skill_id=skill.skill_id AND
skill.category_id=work_category.category_id")
    return render_template('labourer/View_previous_work.html',data=qry)

@app.route('/Add_bill/<cid>',methods=['get','post'])
def Add_bill(cid):
    if session['lg']!='lin':
        return redirect('/')
    if request.method=="POST":
        amnt=request.form['am']
        db=Db()
        db.update("update bill set bill_amount='"+amnt+"' where booking_id='"+cid+"'")
        return '<script>alert("bill added
successfully");window.location="/view_booking"</script>'

```

```

else:
    return render_template('labourer/add_bill.html')
@app.route('/send_reply_service_user/<did>')
def send_reply_service_user(did):
    if session['lg'] != 'lin':
        return redirect('/')
    return render_template('police_station/send_reply1.html',did=did)
@app.route('/send_reply_post_sevice_user',methods=['post'])
def send_reply_post_sevice_user():
    if session['lg']!= 'lin':
        return redirect('/')
    # if request.method=="POST":
    db=Db()
    hid=request.form['did']
    reply=request.form['textarea']
    db.update("update complaints set reply='"+reply+"'where complaint_id='"+hid+"'")
    return '<script>alert("reply sented
successfully");window.location="/view_user_complaints_service"</script>'
    # else:
    #     return render_template("admin/send_reply.html")
@app.route('/user_view_reviews',methods=['post'])
def user_view_reviews():
    db = Db()
    qry = db.select("select * from review,user where review.user_id=user.user_id")
    return render_template('labourer/')
# -----
@app.route('/logout')
def logout():
    if session['lg']!= 'lin':
        return redirect('/')
    session.clear()
    session['lg']=" "
    return redirect('/')
#
=====
=====
=====
#
#
#
=====
=====
=====
=====
#
#
#
=====
=====
=====
=====

```

```

e=request.form['email']
ph=request.form['phonenumber']
hn=request.form['housename']
pl=request.form['place']
po=request.form['post']
ps=request.form['password']
db = Db()
qry=db.insert("insert into login VALUES ('"+e+"','"+ps+"','user')")
db.insert("insert into user VALUES
('"+str(qry)+"','"+n+"','"+e+"','"+ph+"','"+hn+"','"+pl+"','"+po+"')")
return jsonify(status="ok")
@app.route('/and_login',methods=['post'])
def and_login():
    db=Db()
    Username = request.form['u']
    password = request.form['p']
    qry = db.selectOne("select * from login where username='"+ Username + "' and
password='"+ password + "'")
    if qry is not None:
        return jsonify(status="ok",lid=qry['login_id'],type=qry['usertype'])
    else:
        return jsonify(status="no")
@app.route('/and_view_profile',methods=['post'])
def and_view_profile():
    lid=request.form['id']
    db = Db()
    qry=db.selectOne("select * from user where user_id='"+lid+"'")
    return jsonify(status="ok",data=qry)
@app.route('/and_view_workcategory',methods=['post'])
def and_view_workcategory():
    db = Db()
    qry = db.select("select * from work_category")
    if len(qry)>0:
        return jsonify(status="ok",data=qry)
    else:
        return jsonify(status="no")
@app.route('/and_view_laboures',methods=['post'])
def and_view_laboures():
    db = Db()
    qry = db.select("select * from labourer,skill,work_category where
labourer.labourer_id=skill.labourer_id and
skill.category_id=work_category.category_id ")
    if len(qry)>0:
        return jsonify(status="ok",data=qry)
    else:
        return jsonify(status="no")

```



```

@app.route('/and_SCAN_QR',methods=['post'])
def and_SCAN_QR():
    db = Db()
    return jsonify(status="ok")
@app.route('/and_book_worker',methods=['post'])
def and_book_worker():
    db = Db()
    # db.insert("insert into booking VALUES
(", "+booking_id+", "+skill_id+", "+user_id+", "+work_date+", "+work_details+", "+
+booking_date+", "+booking_time+", "+booking_status+")")
    return jsonify(status="ok")
@app.route('/and_view_bill',methods=['post'])
def and_view_bill():
    db = Db()
    qry = db.select("select * from bill")
    return jsonify(status="ok",data=qry)

@app.route('/and_payment',methods=['post'])
def and_payment():
    db = Db()
    # db.insert("insert into payment VALUES (" +payment_id +", "+bill_id +", "+
+payment_date +", "+payment_time +", "+acc_no +", "+amount+")")
    return jsonify(status="ok")
@app.route('/and_send_review',methods=['post'])
def and_send_review():

    lid=request.form['id']
    lbid=request.form['lbid']
    rev=request.form['r']
    db = Db()
    db.insert("insert into review VALUES (,curdate(), "+lid +", "+rev+", "+lbid+")")
    return jsonify(status="ok")
@app.route('/and_view_reviews',methods=['post'])
def and_view_reviews():
    lbid=request.form['lbid']
    db = Db()
    qry = db.select("select * from review,user where review.user_id=user.user_id and
review.lbid="+lbid+"")
    if len(qry)>0:
        return jsonify(status="ok",data=qry)
    else:
        return jsonify(status="no")
@app.route('/view_work_category',methods=['post'])
def and_view_work_category():

```

```

db = Db()
qry = db.select("select * from work_category")
return jsonify(status="ok",data=qry)

@app.route('/and_view_booking',methods=['post'])
def custom_view_booking():
    db = Db()
    lid=request.form['id']
    qry = db.select("select * from work_category inner join skill on
work_category.category_id=skill.category_id inner join labourer on
labourer.labourer_id=skill.labourer_id left join booking on
booking.skill_id=skill.skill_id left join bill on booking.booking_id=bill.booking_id
where booking.user_id='"+lid+"'")
    print(qry)
    if len(qry) > 0:
        return jsonify(status="ok", data=qry)
    else:
        return jsonify(status="no")
@app.route('/booking',methods=['post'])
def booking():
    date=request.form['date']
    lid=request.form['id']
    sk=request.form['skid']
    dtails=request.form['dtls']
    print(date,lid,sk,dtails)
    db=Db()
    qry=db.insert("insert into booking VALUES
('"+sk+"','"+lid+"','"+date+"','"+dtails+"',curdate(),curtime(),'pending')")
    db.insert("insert into bill VALUES ('"+str(qry)+"','0','pending')")
    return jsonify(status="ok")
@app.route('/and_view_more',methods=['post'])
def and_view_more():
    lid=request.form['lbrid']
    db = Db()
    qry=db.selectOne("select * from labourer where labourer_id='"+lid+"'")
    return jsonify(status="ok",data=qry)
@app.route('/and_send_feedback',methods=['post'])
def and_send_feedback():
    lid=request.form['id']
    rev=request.form['r']
    db = Db()
    db.insert("insert into feedback VALUES ('"+lid+"',curdate(),'"+rev+"')")
    return jsonify(status="ok")

```

```

@app.route('/and_custom_view_previous_booking',methods=['post'])
def and_custom_view_previous_booking():
    db = Db()
    lid=request.form['id']
    qry = db.select("select * from booking left join skill on
booking.skill_id=skill.skill_id left join work_category on
skill.category_id=work_category.category_id left join bill on
bill.booking_id=booking.booking_id where booking.user_id='"+lid+"' and
booking.work_date<curdate()")
    print(qry)
    if len(qry) > 0:
        return jsonify(status="ok", data=qry)
    else:
        return jsonify(status="no")
@app.route('/and_offline_payment',methods=['post'])
def and_offline_payment():

    bid=request.form['bid']
    db = Db()
    db.update("update bill set bill_status='offline' where bill_id='"+bid+"'")
    return jsonify(status="ok")
@app.route('/and_online_payment',methods=['post'])
def and_online_payment():
    bid=request.form['bid']
    lid=request.form['lid']
    amnt=request.form['am']
    bn=request.form['name']
    ac=request.form['account']
    ifsc=request.form['ifsc']
    wid=request.form['wid']
    print(lid)
    db = Db()
    qry=db.selectOne("select * from bank where bank_name='"+bn+"' and
ac_no='"+ac+"' and ifsc='"+ifsc+"' and user_id='"+lid+"'")
    print(qry)
    if qry is not None:
        blnc=qry['balance']
        if float(blnc) <= float(amnt):
            return jsonify(status="insuff")
        else:
            db.update("update bank set balance=balance-"+amnt+" where
user_id='"+lid+"'")
            db.update("update bank set balance=balance-"+amnt+" where
user_id='"+wid+"'")
            db.update("update bill set bill_status='paid' where bill_id='"+bid+"'")
            return jsonify(status="ok")

```

```

else:
    return jsonify(status="no")
@app.route('/and_send_complaint',methods=['post'])
def and_send_complaint():
    lid=request.form['id']
    rev=request.form['r']
    db = Db()
    db.insert("insert into complaints_solutions VALUES (',curdate(),'"+lid+"','"
+rev+"','pending','pending')")
    return jsonify(status="ok")
@app.route("/and_view_reply", methods=['post'])
def and_view_reply():
    lid=request.form['lid']
    db=Db()
    res=db.select("select * from complaints_solutions where user_id='"+lid+"'")
    return jsonify(status="ok", data=res)
@app.route("/and_delete_complaint", methods=['post'])
def and_delete_complaint():
    cid=request.form['compid']
    db=Db()
    db.delete("delete from complaints_solutions where complaint_id='"+cid+"'")
    return jsonify(status="ok")
@app.route("/and_view_policestation", methods=['post'])
def and_view_policestation():
    db = Db()
    res=db.select("select * from police_station ")
    return jsonify(status="ok", data=res)
@app.route('/and_send_pcomplaint',methods=['post'])
def and_send_pcomplaint():
    lid=request.form['id']
    rev=request.form['r']
    sid=request.form['sid']
    db = Db()
    db.insert("insert into complaints VALUES
(',"+rev+"',curdate(),'"+lid+"','pending','"+sid+"')")
    return jsonify(status="ok")

@app.route("/and_view_preply", methods=['post'])
def and_view_preply():
    lid=request.form['lid']
    sid=request.form['sid']
    db=Db()
    res=db.select("select * from complaints where user_id='"+lid+"' and
station_id='"+sid+"'")
    return jsonify(status="ok", data=res)

```

```

@app.route("/and_delete_pcomplaint", methods=['post'])
def and_delete_pcomplaint():
    cid=request.form['compid']
    db=Db()
    db.delete("delete from complaints where complaint_id='"+cid+"'")
    return jsonify(status="ok")

# =====qr code
section=====

@app.route('/view_crime_labourer',methods=['post'])
def view_crime_labourer():
    cid=request.form['cid']
    db = Db()
    qry=db.selectOne("select * from labourer where labourer_id='"+cid+"'")
    qry2 = db.select(
        "select * from crime_details,criminals where crime_details.crime_id =
        criminals.crime_id and criminals.labour_id='"+cid+"'")
    return
    jsonify(status="ok",labourer_name=qry['labourer_name'],labourer_email=qry['labour
    er_email'],labourer_photo=qry['labourer_photo'], data2=qry2)
@app.route('/view_crime_details',methods=['post'])
def view_crime_details():
    cid=request.form['cid']
    db = Db()
    qry=db.select("select * from crime_details,criminals where crime_details.crime_id
    and criminals.crime_id and criminals.labourer_id='"+cid+"'")
    if len(qry)>0:
        return jsonify(status="ok",data=qry)
    else:
        return jsonify(status="no")
if __name__ == '__main__':
    app.run(host="0.0.0.0")
    # app.run()

```

## 5.2 user page

### Android Manifet.xml

```

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.safekerala">

    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/iconapp"
        android:label="@string/app_name"

```

```

android:roundIcon="@drawable/iconapp"
android:supportsRtl="true"
android:theme="@style/Theme.SafeKerala"
android:usesCleartextTraffic="true">
<activity
    android:name=".send_pcomplaints"
    android:exported="false" />
<activity
    android:name=".view_preply"
    android:exported="false" />
<activity
    android:name=".custom_view_police_station"
    android:exported="false" />
<activity
    android:name=".view_police_station"
    android:exported="false" />
<activity
    android:name=".custom_view_reply"
    android:exported="false" />
<activity
    android:name=".view_reply"
    android:exported="false"
    android:label="@string/title_activity_view_reply"
    android:theme="@style/Theme.SafeKerala.NoActionBar" />
<activity
    android:name=".send_complaint"
    android:exported="false" />
<activity
    android:name=".custom_view_crime_details"
    android:exported="false" />
<activity
    android:name=".view_crime_details"
    android:exported="false" />
<activity
    android:name=".scanqr"
    android:exported="false" />
<activity
    android:name=".UserHome"
    android:exported="false"
    android:label="@string/title_activity_user_home"
    android:theme="@style/Theme.SafeKerala.NoActionBar" />
<activity
    android:name=".bank_payment"
    android:exported="false" />
<activity
    android:name=".payment_option"

```

```

        android:exported="false" />
<activity
    android:name=".custom_view_previous_booking"
    android:exported="false" />
<activity
    android:name=".view_previous_booking"
    android:exported="false" />
<activity
    android:name=".send_feedback"
    android:exported="false" />
<activity
    android:name=".add_booking"
    android:exported="false" />
<activity
    android:name=".home"
    android:exported="false" />
<activity
    android:name=".user_registration"
    android:exported="false" />
<activity
    android:name=".view_more"
    android:exported="false" />
<activity
    android:name=".custom_view_labourer"
    android:exported="false" />
<activity
    android:name=".custom_view_category"
    android:exported="false" />
<activity
    android:name=".custom_view_booking"
    android:exported="false" />
<activity
    android:name=".custom_view_reviews"
    android:exported="false" />
<activity
    android:name=".view_review"
    android:exported="false" />
<activity
    android:name=".send_review"
    android:exported="false" />
<activity
    android:name=".view_booking"
    android:exported="false" />
<activity
    android:name=".view_labourer"
    android:exported="false" />

```

```

<activity
    android:name=".view_work_category"
    android:exported="false" />
<activity
    android:name=".view_profile"
    android:exported="false" />
<activity
    android:name=".Registration"
    android:exported="false" />
<activity
    android:name=".login"
    android:exported="false" />
<activity
    android:name=".ip_page"
    android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER"/>
    </intent-filter>
</activity>
<activity
    android:name=".MainActivity"
    android:exported="true" />

    <uses-library android:name="org.apache.http.legacy" />
</application>
</manifest>

```



## **6. SYSTEM TESTING**

## 6.1 TESTING AND EVALUATION

Testing is a process of executing a program with the intent of finding an error. Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design and coding. Testing includes verifications of the basic logic of each program and verification that the entire system works properly. Testing demonstrates that software functions appear to be working according to specification. In addition, data collected as testing is conducted provides a good indication of software quality as a whole. The debugging process is the most unpredictable part of testing process. Testing begins at the module level and works towards the integration of the entire computer-based system testing and debugging are different activities, but any testing includes debugging strategy for software testing must accommodate low level tests that are necessary to verify that a small source code segment has been currently implemented as well as high-level tests that validate major system function, against customer requirements. No testing is complete without verification and validation part. The goals of verification and validation activities are to access and improve the quality of work products generated during the development and modification of the software. There are two types of verification: life cycle verification and formal verification. Life cycle verification is the process of determining the degree to which the products of the given phase of the development cycle fulfil the specification established during the prior process. Formal verification is the rigorous mathematical demonstration that source code conforms to its specifications. Validation is a process of evaluating software at the end of the software development process to determine conformance with the requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The primary objectives, when we test software are the following:

- Testing is a process of executing with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.
- A successful test is one uncovers undiscovered errors.

Thus, testing plays a very critical role in determining the reliability and efficiency of the software and hence is very important stage in software development. Tests are

to be conducted on the software to evaluate its performance under a number of conditions. Ideally, it should so at the level

of each module and also when all of them are integrated to form the completed system. Software testing is done at different levels.

## **6.2 TESTING STRATEGIES**

A strategy for software testing integrates software test case design method in to a well-planned series of steps that result in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote a customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project progresses.

**The general characteristics of software testing strategies are:**

- Testing begins at the component level and works “outward” toward the integration of the entire computer system.
- Different testing techniques are appropriate at different points in time.

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

A strategy must provide guidance for the practitioner and set of milestones for the manager. Because the step on the test strategy occurs at a time when deadline pressure begins to rise, progress must be measurable and problem must surface as early as possible.

The software team’s approach to testing is defining a plan that describes an overall strategy and a procedure that defines specific testing steps and tests that will be conducted. In the proposed system, if the administrator makes any attempt to login to the application without entering his password, then the system will not allow the user to login to the application.

## **6.3 TESTING TECHNIQUES**

The various testing techniques are given below:

### **6.3.1 WHITE BOX TESTING**

White-box testing is also called as glass-box testing, is a test case design method that goes to the control structure of the procedural design to derive

test cases. Using white box testing methods, the software engineer can derive test cases that,

- ✓ Guarantee that all independent paths within a module have been exercised at Least once.
- ✓ Exercise all logical decision on their true and false sides.
- ✓ Execute all loops at their boundaries and within their operational sides.
- ✓ Exercise internal data structure to ensure their validity.

White box testing was successfully conducted on our system. All independent paths within a module have been executed at least once and all logical decisions have been exercised on their true and false sides.

### **6.3.2 BLACK BOX TESTING**

Black-box testing is also called as behavioural testing, focuses on the functional requirement of the software. It is a complimentary approach that is likely uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories.

- ✓ Incorrect or missing functions.
- ✓ Interface errors.
- ✓ Error on data structures or external database access.
- ✓ Behaviour or performance errors.
- ✓ Initialization and termination errors.

Black box testing was successfully conducted on your system. The system was divided into a number of modules and testing was conducted on each module. We have tested the system for incorrect or missing functions, interface and performance errors.

### **6.3.3 UNIT TESTING**

Unit testing comprises the set of tests performed by an individual programmer prior to the integration of the system. Testing removes residual bugs and improves the reliability of the system.

Testing allows the developer to find out the design faults if any, and enable correction if needed. Exhaustive unit testing has to be carried out to ensure the validity of the data. In order to successfully test the entire package, unit testing is carried out. Each module was tested as when it was developed. Thus it proved easier to conduct minute testing operation and correct them then and there.

### **6.3.4 INTEGRATION TESTING**

Bottom-up integration is the traditional strategy used to integrate the component of a software system into a functional whole. Bottom-up integration consists of unit testing, followed by subsystem testing and followed by testing of the entire system. Unit testing has the goal of discovering errors in the individual parts of the system.

Parts are tested in isolation from one another in an artificial environment known as “Test Harness”, which consists of driver programs and data necessary to exercise the modules. Unit testing should be as exhaustive as possible to ensure that each representative case handled by each module has been tested. Unit testing is eased by a system structure that is composed of small loosely coupled modules.

Both control and data interfaces must be tested. Large software system may require several levels of subsystem testing. Lower level subsystems are successively combined to form higher level subsystems. In most software systems, exhaustive testing of subsystem capabilities is not feasible due to the combination complexity of the module interfaces. Therefore, test cases must be carefully chosen to exercise the interfaces in the desired manner.

### **6.3.5 ACCEPTANCE TESTING**

Acceptance testing involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements. It is not unusual for two sets of acceptance test to be run, those developed by the quality group and those developed by the customer.

In addition to the functional and performance tests, stress tests are performed to determine the limitation of the system. For example, a compiler might be tested to determine the effect of the symbol table overflow, or real-time system might be tested to determine the effect of simultaneous arrival of numerous high priority interrupts.

#### **6.3.6 OUTPUT TESTING**

Output testing of the proposed system is important since no system could be useful if it does not produce the required output.

The output format on the screen is found to be correct, as the format was designed in the system design phase according to the user needs. For the hard copy also the output comes out as the specified requirements by

the user. Hence output testing doesn't result in any correction on the system.

## **7.SYSTEM IMPLEMENTATION AND DEPLOYMENT**

Implementation is the process of deploying the new system in the operational environment. Proper implementation is essential to provide a reliable system to meet the organizational requirement. There are four types of implementation methods. They are Direct Changeover, Phased Implementation, Parallel Run and Pilot Approach. The most commonly used implementation methods are Pilot Approach and Parallel Run.

The system which is developed as a web application hence the other functions as normal application, as usual some web development technologies are used in the implementation of the project. The language I selected to program this software is PHP. The reason I selected PHP is that is a simple and powerful language that especially developed to create web application.

Technologies used in the development of the software are:

- ✓ Programming language – Python
- ✓ DBMS – MySQL server
- ✓ Development tool – Py charm
- ✓ Development platform – Windows 10

The front end is HTML, and CSS and back end is MySQL Server and Python. The system developed on Pycharm in Windows 10 operating system.



## **8. CONCLUSION**

The “SAFE KERALA” has been developed for all given conditions and it is found working effectively under the all the circumstances that may arise in the real environment. The software has been developed to reducing the operator work.

This system is user friendly and is well efficient to make easy interaction with the users of system. The system is done with an insight into the necessary modifications that may be required in the future. Hence the system can be maintained successfully without much work.

## **8.1 FUTURE ENHANCEMENT**

As a future venture, it is suggested to make some changes to provide more services and to provide information at right time in right manner.

The current system is designed to manage only migrated labours but the future enhancement is to provide for new labours around every field . in the future it can also be used for jobs and studys around the globe

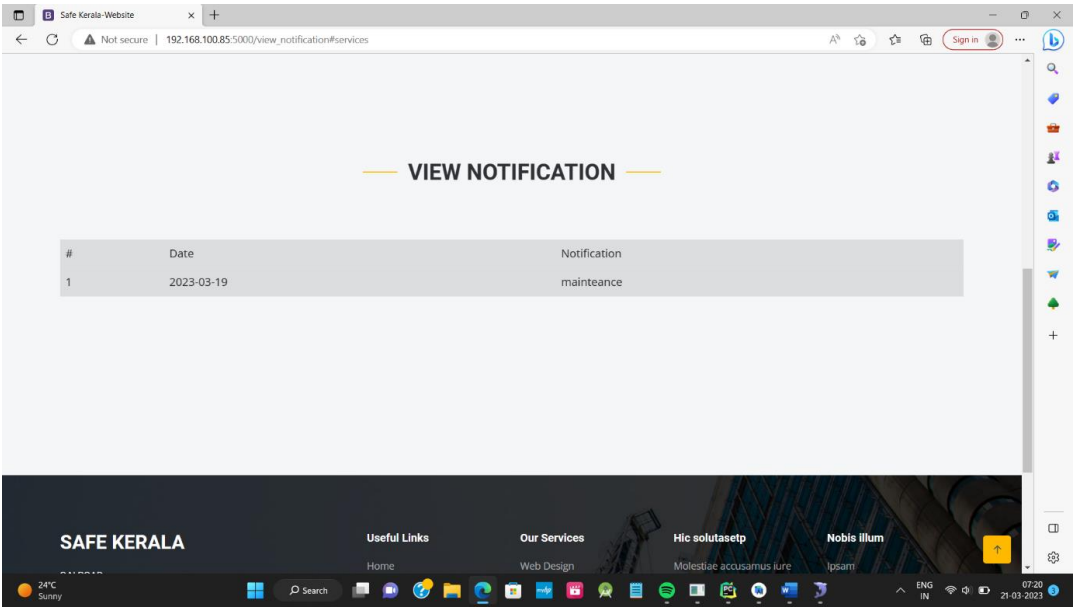
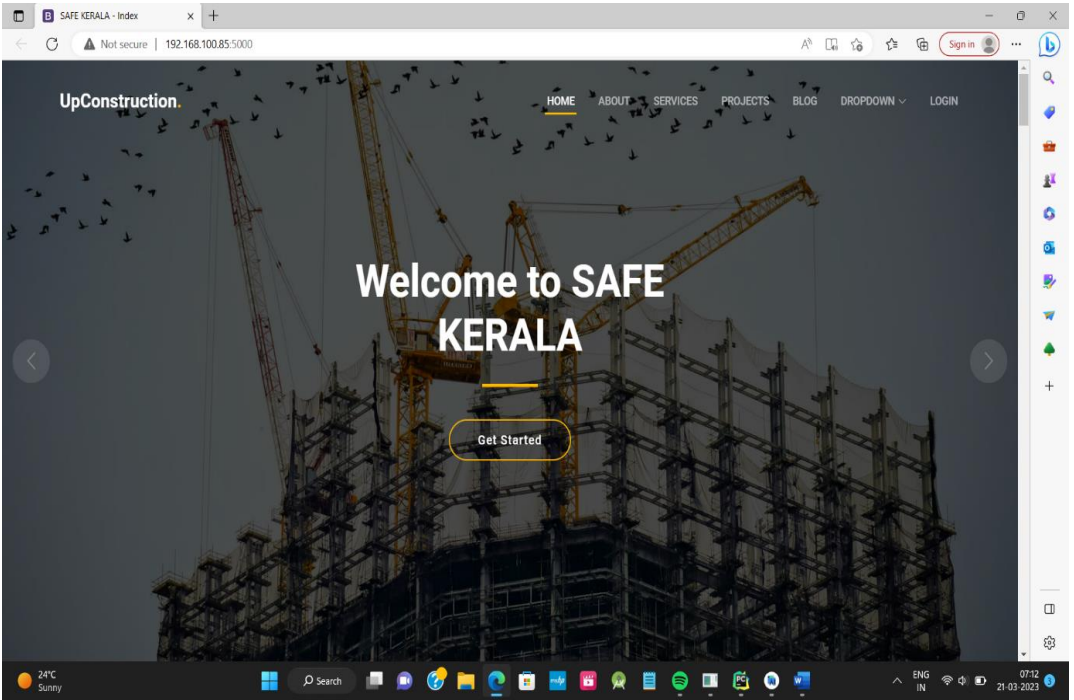
## 9. REFERENCE

- Websites:
  - w3school.com
  - ol.com
  - tutorialspoint.com
- YouTube
- Books:
  - **Database System Concept: Marvin .F. Korth**
  - **Programming With Java : A Primer, E Balaguruswami, Tata Mc Graw Hill,2008 Professional Android 4, Satya Komatneni &Dev MacLean**
  - **Apress Software Engineering : Rajib Mall**

## **10. APPENDIX**

# 10.1 Screenshots

## 10.1.1 Homepage:



Safe Kerala-Website x +

Not secure | 192.168.100.85:5000/view\_feedback\_service#services

Sign in

## VIEW FEEDBACK

#	Username	Date	feedback
1	Naveen john	2023-03-20	good

**SAFE KERALA**

Useful Links: Home, Web Design, Molestiae accusamus iure, Ipsam

Our Services: Hic solutasetp, Nobis illum

24°C Sunny

ENG IN 07:20 21-03-2023

Safe Kerala-Website x +

Not secure | 192.168.100.85:5000/view\_user\_complaints\_service#services

Sign in

## VIEW USER COMPLAINTS

#	date	User_info	complaints	Reply
1	2023-03-20	Naveen john	theft	ok.. complaint registered. will contact you as soon as possible

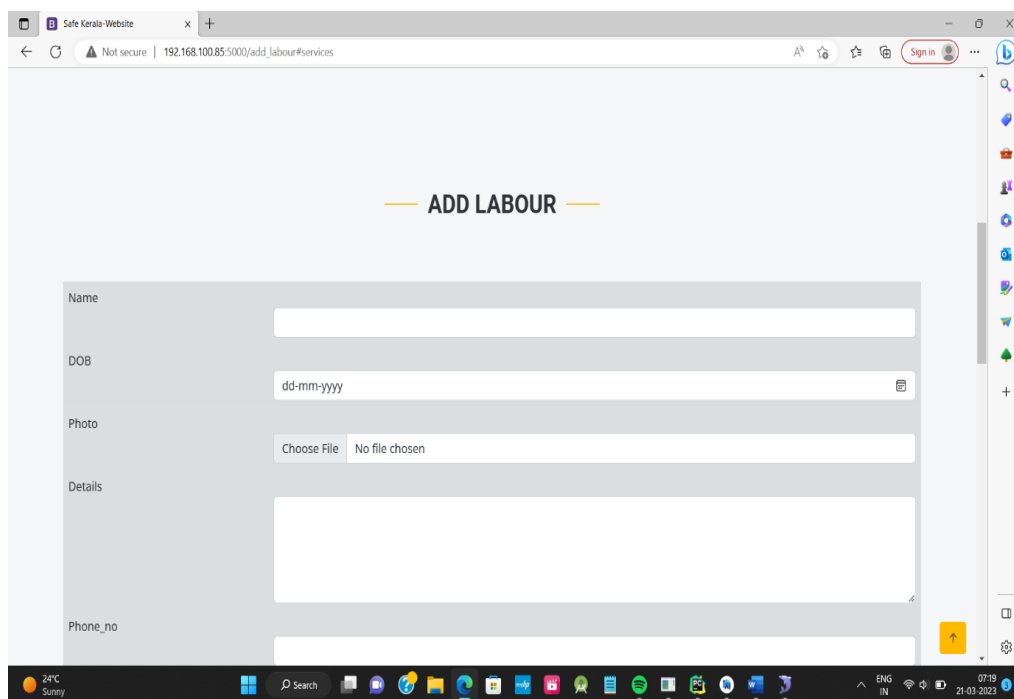
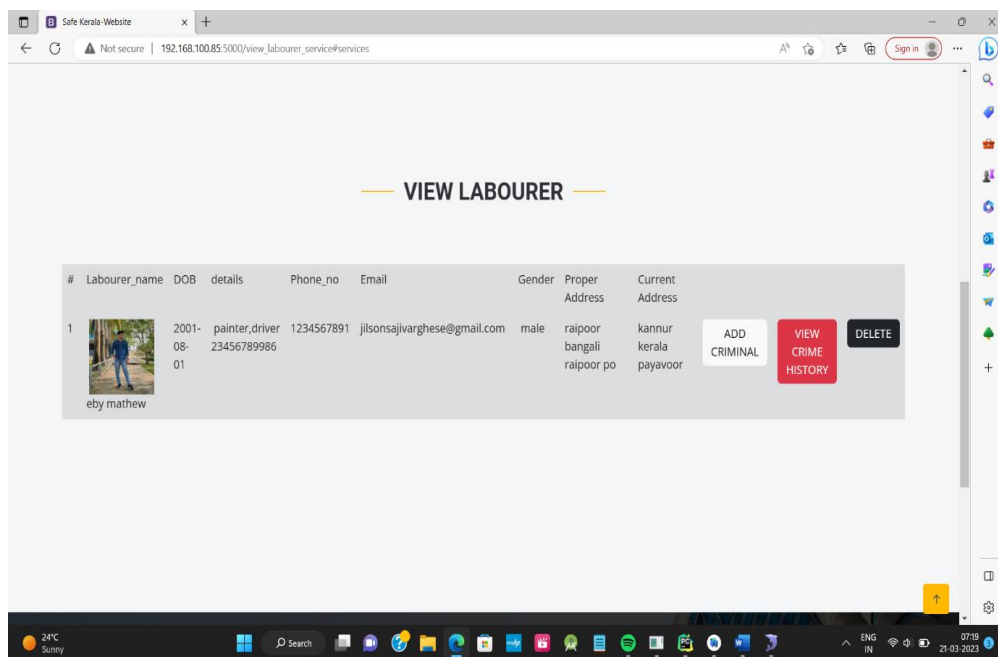
**SAFE KERALA**

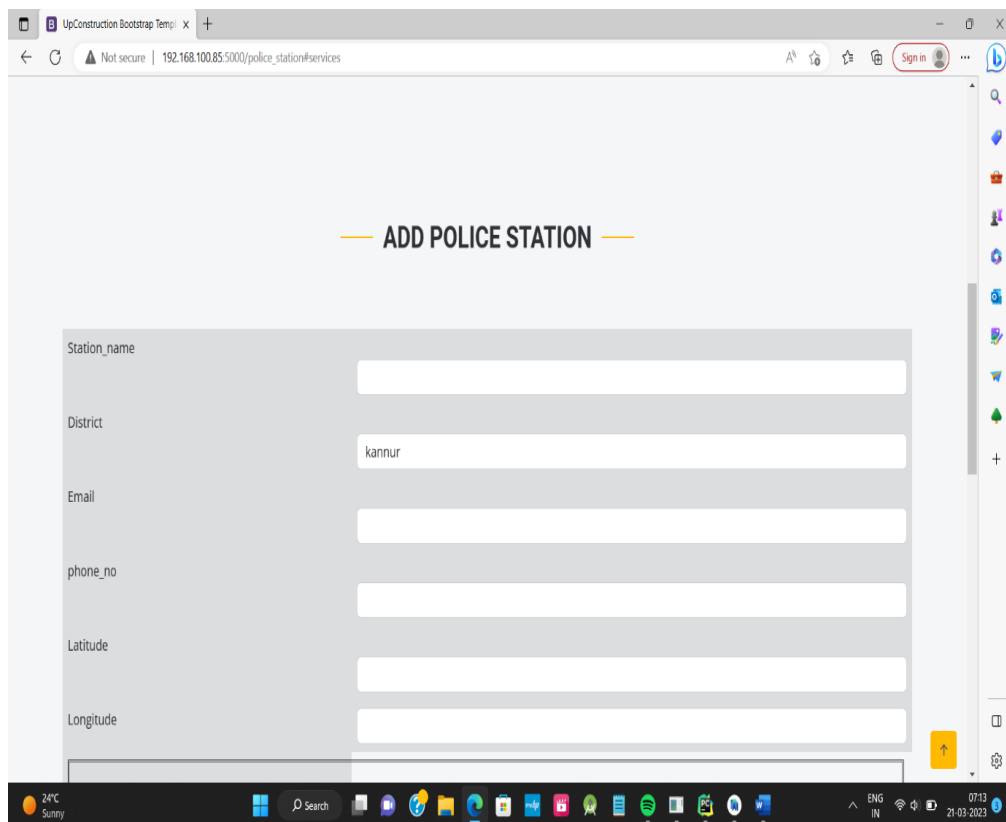
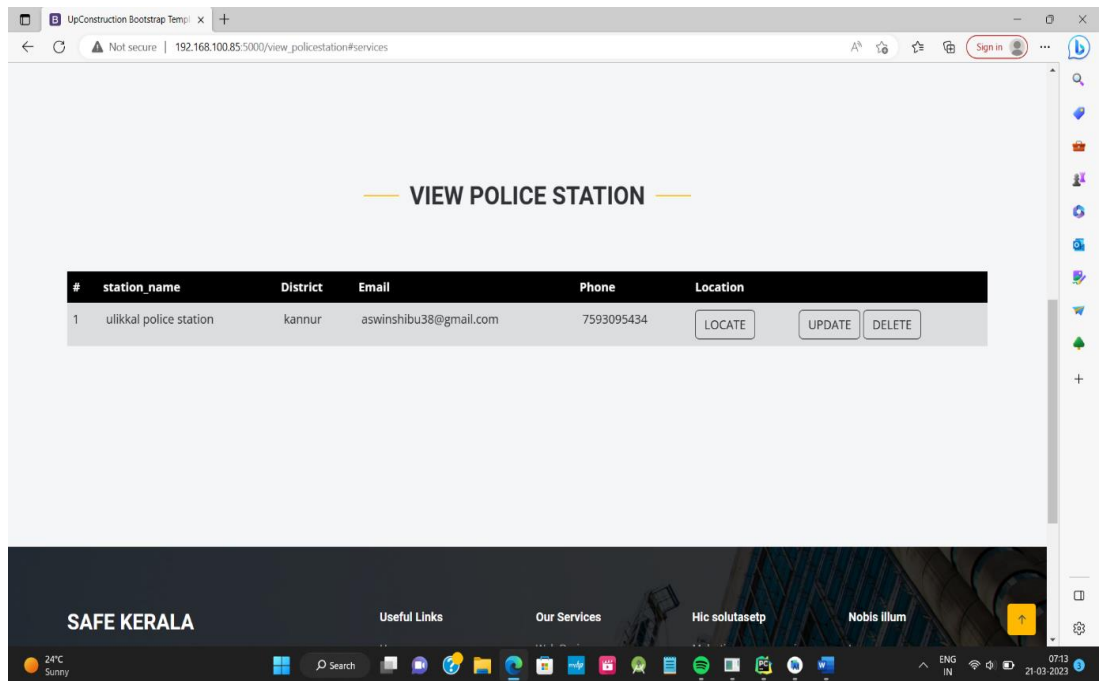
Useful Links: Home, Web Design, Molestiae accusamus iure, Ipsam

Our Services: Hic solutasetp, Nobis illum

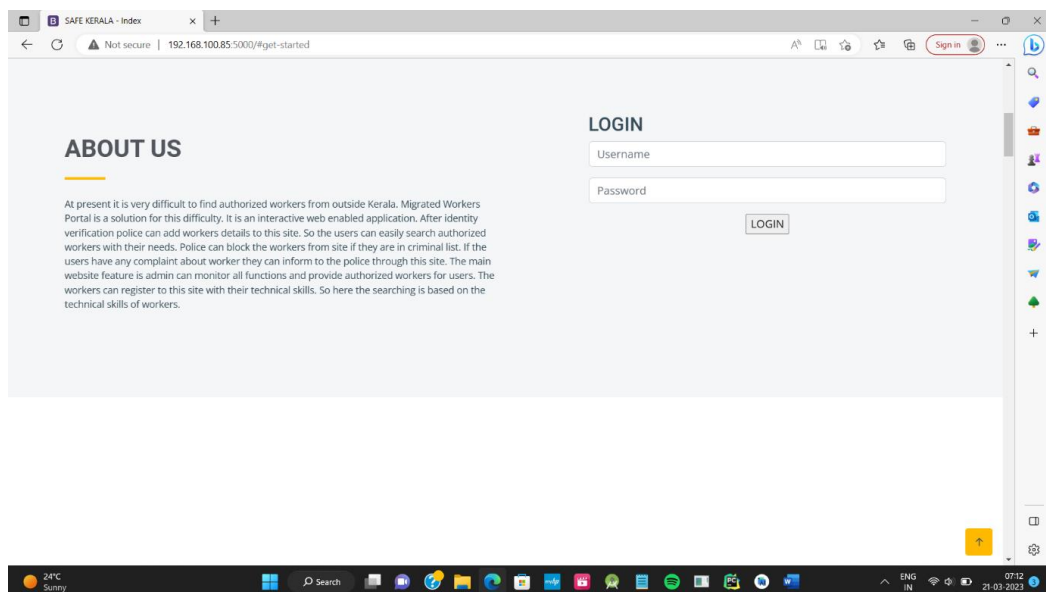
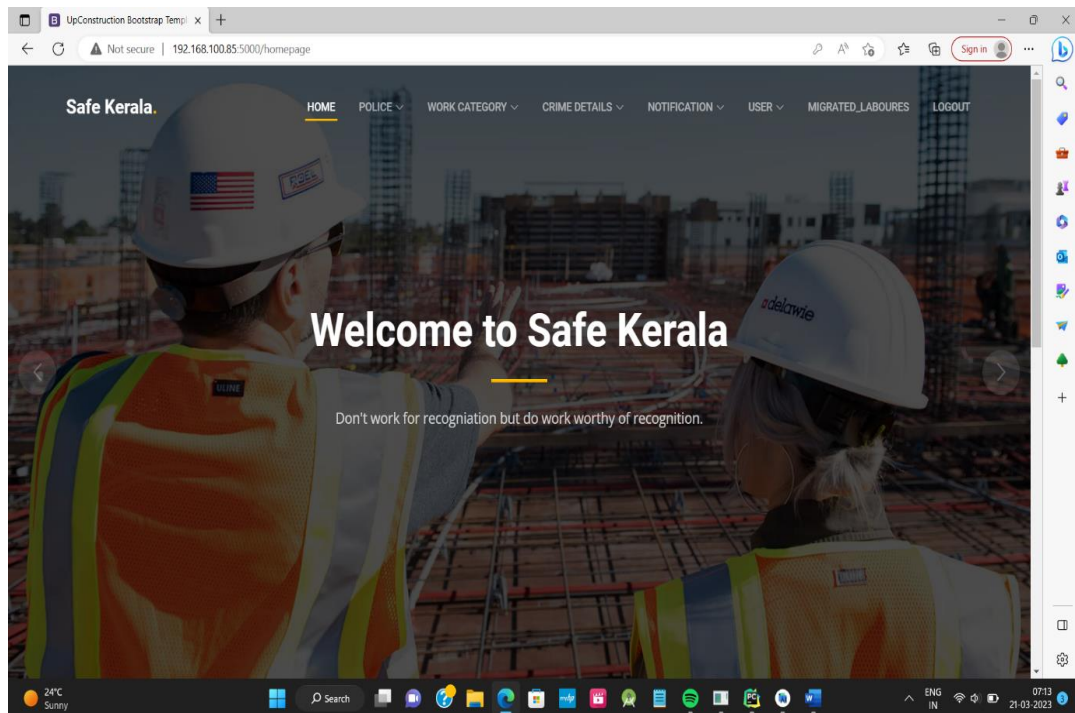
24°C Sunny

ENG IN 07:19 21-03-2023









10:50 AM | 0.1KB/s

VoWiFi 76

Safe Kerala

enter ip

ENTER



10:51 AM | 0.1KB/s

Vo WiFi 76

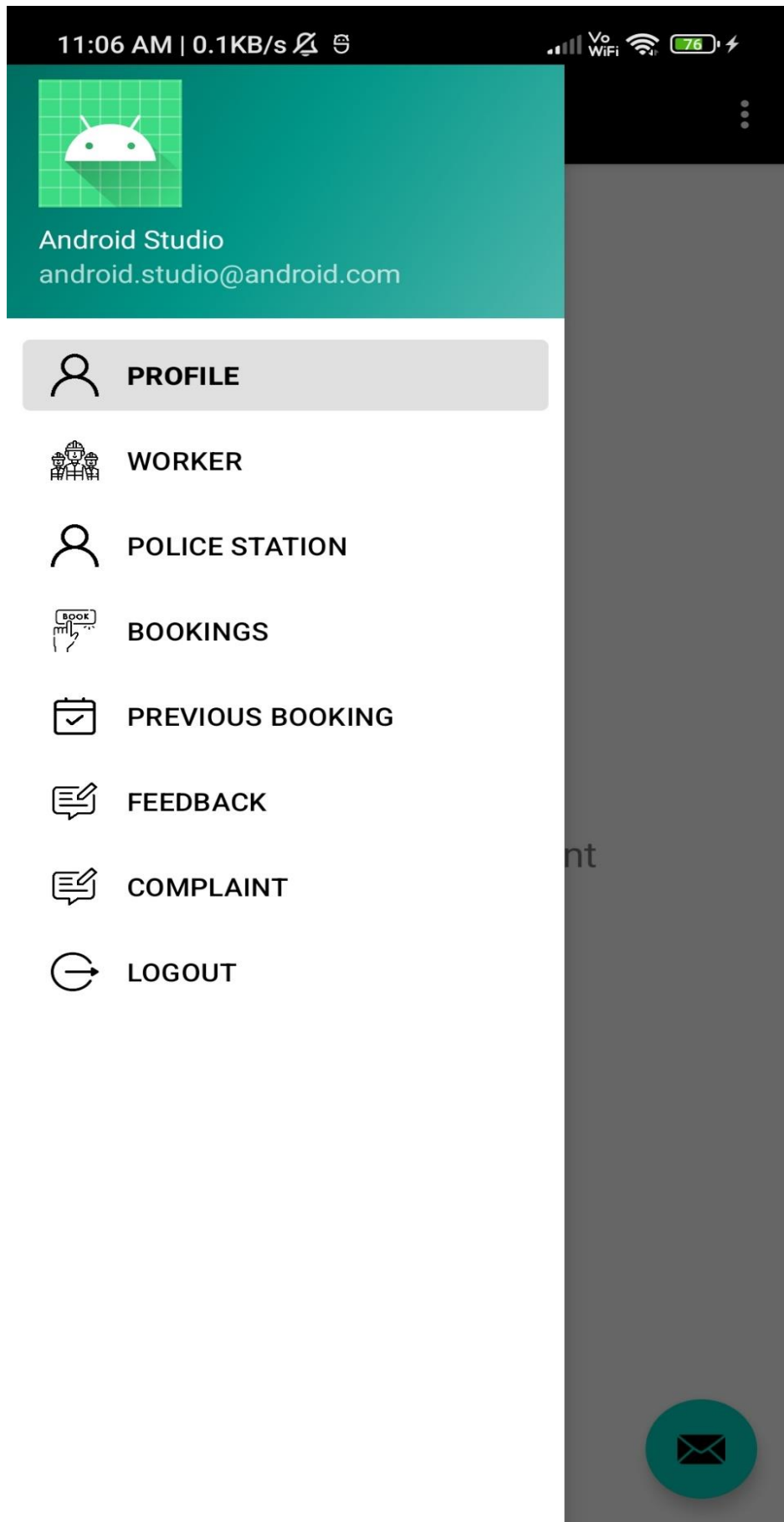
Safe Kerala

Username

Password

LOGIN

New user?



## Safe Kerala

### MY PROFILE

Name	Naveen john
Email	s
phone	7593095434
house name	mangalathu
place	ulikkal
post	ulikkal

## Safe Kerala



Labourer name **eby mathew**  
Aadhar Number 23456789986  
DOB painter,driver  
details bangali  
Address 1234567891  
Phone jilsonsajivarg  
hese@gmail  
.com  
Email carpenter  
category 2001-08-01

BOOK

REVIEW

## Safe Kerala

Station name **ulikkal police station**

District kannur

Email aswinshibu38@gmail.com

Phone number 7593095434

SEND COMPLAINT

## Safe Kerala

date 2023-3-31

amount 500

worker name eby mathew

worker details painter

VIEW MORE

PAYMENT

REVIEW

date 2023-3-30

amount 0

worker name eby mathew

worker details driver

VIEW MORE

REVIEW



## Safe Kerala

date 2023-03-19

amount 2000

worker name null

worker details gg

VIEW MORE

date 2023-3-31

amount 500

worker name carpenter

worker details painter

VIEW MORE

date 2023-3-30

amount 0

11:12 AM | 0.4KB/s



Safe Kerala

FEEDBACK

Feedback

SEND

## view\_reply

date	Mon, 20 Mar 2023 00:00:00 GM
Complaint	bad
Reply Date	null
Reply	pending

DELETE



# **A PROJECT REPORT ON WOLVERINE**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ANUMOD C**

**REG.NO: DB20BCAR06**

**ALAN ALIAS**

**REG.NO: DB20BCAR03**

**ADARSH MV**

**REG.NO: DB20BCAR01**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**

# **A PROJECT REPORT ON WOLVERINE**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ANUMOD C**

**REG.NO: DB20BCAR06**

**ALAN ALIAS**

**REG.NO: DB20BCAR03**

**ADARSH MV**

**REG.NO: DB20BCAR01**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**

**DON BOSCO ARTS & SCIENCE COLLEGE**  
**ANGADIKADAVU**  
**IRITTY, KANNUR**



**CERTIFICATE**

*Certified that this report titled **WOLVERINE** is a bonafide record of the project work done by Anumod c(Reg.No: DB20BCAR06) , Alan Alias (Reg.No:DB20BCAR03) and Adarsh MV (Reg.No:DB20BCAR01) under the supervision and guidance ,towards partial fulfilment of the requirement for award of the degree of bachelor of computer application (BCA) of the Kannur university.*

Project Guide

Head of the Department

Angadikadavu

External Examiner

Date:

- 1.
- 2.

## Declaration

We ANUMOD C, ALAN ALIAS and ADARSH MV sixth semester BCA student of Don Bosco Arts & Science College, Angadikadavu, under Kannur University do hereby declare that the project entitled **“WOLVERINE”** is the original work carried out by me in the sixth semester under the supervision of Ms.Sindu PM, Lecture of the Department of BCA, Don Bosco Arts & Science College, Angadikadavu, in partial fulfilment of the requirement for the award of the degree Bachelor of Computer Application, Kannur University.

Angadikadavu

ANUMOD C

ALAN ALIAS

Date

ADARSH MV

## **ACKNOWLEDGEMENT**

First of all we thank the lord almighty for his immense grace and blessings showered on me at every stages of this work. I am greatly indebted to our Principal Fr. Dr. Francis Karackat SDB, Don Bosco Arts & Science College, Angadikadavu for providing the opportunity to take up this project as part of my curriculum.

We deeply indebted to my project guide Ms.Sindu PM, lectures of department of BCA, for her assistance and valuable suggestions as guide. She made this project a reality.

We express our sincere thanks to Mrs. Sruthi Nimesh, Mr. Hebin Layola, Mrs. Fincy Cyriac and Mrs. Vineetha Mathew, lecturers of department of BCA, for their valuable suggestions during the course of this project. Their critical suggestions helped me to improve the project work.

Acknowledging the efforts of everyone, their chivalrous help in the course of the project preparation and their willingness to collaborate with the work, their magnanimity through lucid technical details lead to the successful completion of my project.

We would like to express my sincere thanks to all my friends, colleagues, parents and all those who have directly or indirectly assisted during this work.



# CONTENTS

<b>chapters</b>	<b>contents</b>	<b>Page No</b>
1	Introduction	1
2	System Analysis	7
2.1	Existing System	8
2.1.1	Disadvantage Of Existing System	8
2.2	Proposed System	8
2.2.1	Advantage Of Proposed System	9
2.3	Feasibility Study	9
2.3.1	Economic Feasibility	10
2.3.2	Technical Feasibility	10
2.3.3	Behavioural Feasibility	11
2.4	System Specification	11
2.4.1	Software Specification	11
2.4.2	Hardware Specification	12
2.5	Identification Of Actors	12
2.6	Identification Of Use Cases	13
2.6.1	Use Cases For The Actor Admin	13
2.6.2	Use Cases For The Actor user	14

2.6.3	Use Cases For The Actor traffic police	15
2.6.4	Use Case Diagram	16

3	SYSTEM DESIGN	19
3.1	Introduction	20
3.2	Database Design	20
3.3	Table Design	21
3.4	Tables	22
3.5	Data Flow Diagram	26
3.6	ER Diagram	34
4	CODING	36
4.1	Input Interface	37
4.2	Output Interface	37
4.3	Software Description	37
4.4	HTML	37
4.4.1	CSS	38
4.4.2	JavaScript	39
4.4.3	MySQL	40
4.4.4	Python	43
4.4.5	Flask	58
5	CODING PAGES	59
5.1	Admin Page	60
5.2	Traffic police	72

5.3	Main Section	74
6	System Testing	76

6.1	Testing And Evaluation	77
6.2	Testing Strategies	78
6.3	Testing Techniques	79
6.3.1	White Box Testing	79
6.3.2	Black Box Testing	79
6.3.3	Unit Testing	80
6.3.4	Integration Testing	80
6.3.5	Acceptance Testing	80
6.3.6	Output Testing	81
7	SYSTEM IMPLEMENTATION AND DEPLOYMENT	82
8	CONCLUSION	84
9	REFERANCE	86
10	APPENDIX	88

## **1.INTRODUCTION**

## **PROJECT OVERVIEW:**

In developed countries large-scale technology is used for the monitoring and management of road infrastructure. The WOLVERINE application is implemented for to identify surface disruptions based on accelerometer patterns and also use a crowd sourcing technique for store or pass incident details to the server. Sensor networks together with surveillance cameras help identify elements disturbing cars' mobility. On the contrary, in developing countries, the lack of this level of infrastructure makes the task of roads' maintenance and traffic control challenging. The presence of roadway surface disruptions (RSDs), in particular, affects the economy and reputation of individuals and companies. Urban computing strategies can be adopted to collect data about the presence of RSDs and be applied to monitor traffic related issues. The implementation of such level of navigation assistance could help save freight and money to companies and authorities, and even reduce cars' accidents. In this direction, current mobile technologies can help with the identification and location of road imperfections alerting drivers about alternative routes. In this paper we present our work that seeks to enable citizens' cars as road watchers. By means of the mobile phones' acceleration sensing capabilities we are identifying and tagging the presence of RSDs. Using Android-based devices situated on the copilot floor side of a car, 5 Mbytes of road information has been collected. We run a series of experiments aiming to differentiate acceleration patterns associated to potholes, speed bumps, metal humps and rough roads. Currently, the classification of disruptions is being experimented with techniques from the field of Machine Learning (ML) such as artificial neuronal networks and logistic regression. Classification of individual events is over 86% of accuracy that is competitive with those reported in the literature. In this work we provide the first public dataset that could be used by other researchers to offer more insight in this problem. So this disruption finding is an important feature of this application. Then by using WOLVERINE, user can send incident notification to

the server. But here applying a crowd sourcing technique to check the data is fake or not and based on this result the data will be stored or pass to the server and share the information. Voice command alert is another feature of this application and that means, the user can hear a voice commands about the signal rules. And also the user can view the important areas like hospital, school etc.

### **OBJECTIVE:**

To design an efficient information system for road travelers through smartphone.

### **Motivation or Relevance:**

Monitoring road and traffic conditions in a city is a problem widely studied. Several methods have been proposed towards addressing this problem. Several proposed techniques require dedicated hardware such as GPS devices and accelerometers in vehicles or cameras on roadside and near traffic signals. All such methods are expensive in terms of monetary cost and human effort required. We propose Wolverine - a non-intrusive method that uses sensors present on smartphones. We extend a prior study to improve the algorithm based on using accelerometer, GPS and magnetometer sensor readings for traffic and road conditions detection. We are specifically interested in identifying braking events - frequent braking indicates congested traffic conditions - and bumps on the roads to characterize the type of road.

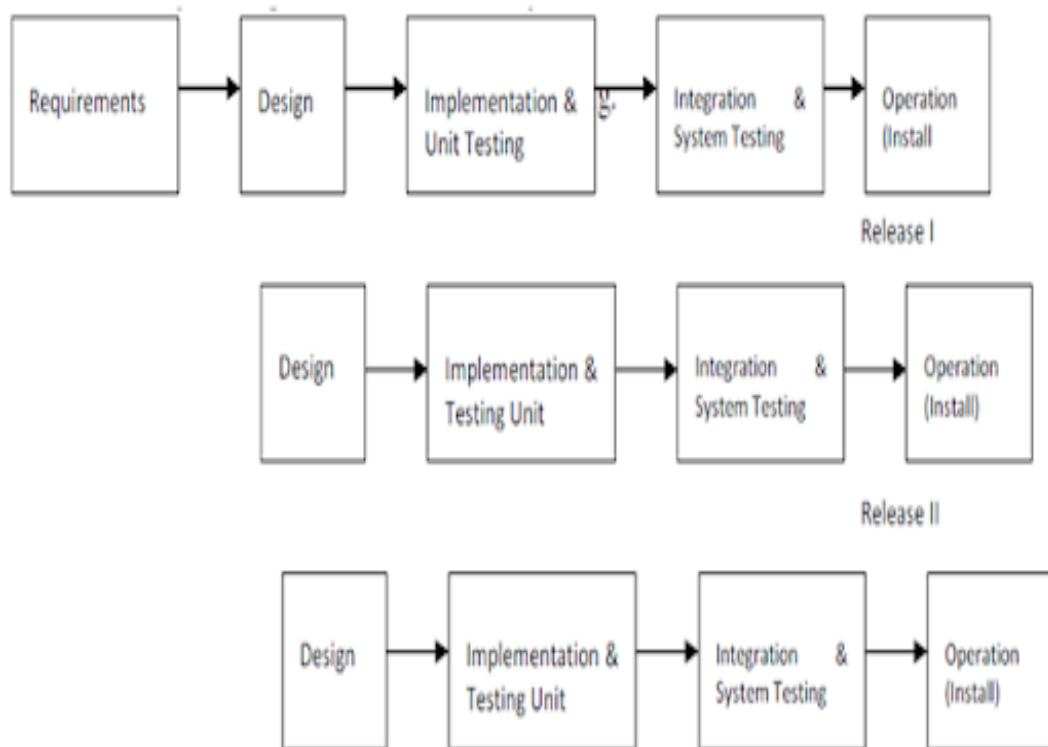
### **PROBLEM DEFINITION:**

Proposed system is a fine combination of Android mobile technology and embedded system. An application should be installed on android mobile handset to get voice notifications about road disruptions. User can send details about traffic signals, accidents and some other disruptions. It includes emergency complaint section to get spot help.

## **MODEL:**

### **Increment process model**

- Increment process models are effective in the situations where requirement are defined precisely and there is no confusion about the functionality of the final product.
- After every cycle, a useable product is given to the customer.
- This model has the same phases as the waterfall model, but with fewer restrictions.
- Generally the phase occur in the same order as in the waterfall model, but these may be conducted in several cycles
- A useable product is released at the end of the each cycle, with each release providing additional functionality.
- During the first requirement analysis phase, customers and developers specify as many requirement as possible and prepare a SRS documents .
- Developers and customers then prioritize these requirements.
- Developers implement the specified requirements in one or more cycles of design, Implementation and test based on the defined priorities
- This model does deliver an operational quality product at each release, but one that satisfies only a subset of the customers requirements
- The complete product is divided into releases and the developer delivers the product release by release.
- At each release customer has an operational quality product that does a portion of what is required
- With this model first release may be available within few weeks or months, whereas the customer generally waits months or years to receive a product using the waterfall and prototyping model



### When we use the Incremental Model?

- When the requirements are superior.
- A project has a lengthy development schedule.
- When Software team are not very well skilled or trained.
- When the customer demands a quick release of the product.
- You can develop prioritized requirements first.

### Advantage of Incremental Model

- Errors are easy to be recognized.
- Easier to test and debug
- More flexible.
- Simple to manage risk because it handled during its iteration.
- The Client gets important functionality early.

### Disadvantage of Incremental Model

- Need for good planning
- Total Cost is high.
- Well defined module interfaces are needed.



## **2. SYSTEM ANALYSIS**

## **INTRODUCTION**

System analysis is the process of collecting and interpreting facts, understanding problems and using the information to suggest improvement on the system. This will help to understand the existing system and determine how computers make its operation more effective. The aim of this analysis is to collect detailed information on the system and the feasibility study of the proposed system.

### **2.1 EXISTING SYSTEM**

Application that charts physical locations on globe for identification and navigation - **Google Map.**

#### **2.1.1 The Disadvantages of Existing System**

- Existing systems gives info about routes and important places only. No systems provide info about road disruptions and traffic signals.
- High Cost

### **2.2 PROPOSED SYSTEM**

#### **2.2.1 Advantages of Proposed System**

The WOLVERINE application is implemented for to identify surface disruptions based on accelerometer patterns and also use a crowd sourcing technique for store or pass incident details to the server. Sensor networks together with surveillance cameras help identify elements disturbing cars' mobility. On the contrary, in developing countries, the lack of this level of infrastructure makes the task of roads' maintenance and traffic control challenging. The presence of roadway surface disruptions (RSDs), in particular, affects the economy and reputation of individuals and companies. Urban computing strategies can be adopted to collect data about the presence of RSDs and be applied to monitor traffic related issues. Then by using WOLVERINE, user can send incident notification to the server. But here applying a crowd sourcing technique to check the data is fake or not and based on this result the data will be stored or pass to the server and share the information. Voice command alert is another feature of this application and that means, the user can hear a voice commands about the signal rules. And also the user can view the important areas like hospital, school etc.

## **2.3 Feasibility Study**

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioral feasibilities.

The proposed system is theoretically investigated to check the feasibility and found that they are more reliable and efficient in the cases given below. There are three aspects in the feasibility study portion of the preliminary investigation.

✓ Economic feasibility

✓ Technical feasibility

✓ Behavioural feasibility

The proposed system must be evaluated from a technical point of view first, and if technical feasible their impact on the organization must be assessed. If compatible, the operational system can be devised. Then they must be tested for economic feasibility.

### **2.3.1 Economic Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors which affect the development of a new system is the cost it would require. Since the system developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

### **2.3.2 Technical Feasibility**

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, procedures and staff. Having identified an outline system, the investigation must go on suggest the type of equipment, required method developing the system, of running the system once it has been designed. The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Though the technology become obsolete after some period of time, due to the fact that newer version of some software supports older versions, the system still be used. So there are only minimal constraints involved with this project. The system has been developed using C# and .NET, along with the database software SQL server, thus we could conclude that the project is technically feasible for development.

### **2.3.3 Behavioural Feasibility**

People are inherently resistant to change and computers have been known to facilitate change. The System is designed in user friendly manner and we need to provide any special training for the persons using this software. The operating system used is Windows 10, which is also user friendly. Since the application is web biased and can easily accessed in a web browser, which is quite familiar to the intended users, it does not have any operational barriers. So no need to provide any special training for using this application software and hence it is behaviourally feasible.

## **2.4 System Specifications**

System Specification deals with the technical aspects the project has to meet in minimum to work successfully. This also includes the different aspects the software requirement is determined from. The technical details typically include:

- Software Specification
- Hardware Specification

### **2.4.1 Software Specifications**

The software required for the application depends on the following factors:

- ✓ The flexibility of the software

- ✓ Software contracts
- ✓ Limitation of the software

### **Software Requirement**

This specifies the minimum software requirements for implementing the system. This includes:

- Front End: - HTML, CSS, Bootstrap
- Back End: - Python 3.9, SQL
- Client side scripting: java script
- Server side scripting: Python
- Platform:-Flask
- Operating System: Microsoft windows 10

### **2.4.2 Hardware Specifications**

The software required for the application depends on the following factors:

- ✓ Determining size and capacity requirements.
- ✓ Computer evaluation and measurement.
- ✓ Financial factors.
- ✓ Maintenance and support.

### **Hardware Requirement**

- Microprocessor: Any 64 bit processor.
- Clock speed: - 2.13GHz
- Ram: 1 GB and above
- Hard disk: 40 GB and above
- Keyboard:-standard keyboard
- Mouse: Standard mouse
- Connectivity: - LAN & Wi-Fi
- Camera: Standard Camera

## **2.5 Identification of Actors**

A use cases represents the functionality of an actor. It is defined as a set of actions performed by a system, which yields an observable result. An ellipse containing its name inside the ellipse or below it represents. it. It is placed inside the system

boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

We can identify the actors through a list of questions. The answers to these questions bring out the actors of the system is.

- Admin
- Traffic
- User

Here we need to specify the use cases of each actor.

## **2.6 Identification of use cases**

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yield an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

To find out the use cases, ask the following questions to each of the actors.

- ✓ Which functions does the actor require from the system? What does the actor need to do?
- ✓ Does the actor need to read, create, destroy, modify or store some kind of information in the system?
- ✓ Does the actor have to calculate something? And want to provide information for others?
- ✓ Could the actor's daily work be simplified or made more efficient by adding new functions to the system (typically functions which are currently not automated in the system)?

### **2.6.1 Use cases for the actor Admin**

#### **Login:**

The first step involved is login. The admin can login to the website using username and password.

#### **Traffic Signal Management:**

View and add traffic signal.

**Traffic Police Management:**

Add and view traffic police.

**Important Places Management:**

View and add important places.

**View Spot Complaint:**

View and add spot complaint.

**2.6.2 Use cases for the actor Trafficpolice**

**Login:**

The trafficpolice can login to the website using username and password.

**View spot complaint:**

Can view spot complaints.

**View emergency alert and point it's source:**

View emergency alert and point it's source using google map.

**Update status:**

Can update status.

**Over speed:**

Can view the over speeding users with their details.

**2.6.3 Use cases for the actor User**

**Registration:**

User can register to the application using a username and password.

**Login:**

User can login to the website using username and password.

**Send emergency alert:**

User can send emergency alerts.

**Send spot complaint:**

User can send spot complaints.

**View important places:**

User can view important places.

**View my position:**

User can view their position

**View emergency alert:**

User can view emergency alerts.

**Send important points:**

User can send important points.

**View signal:**

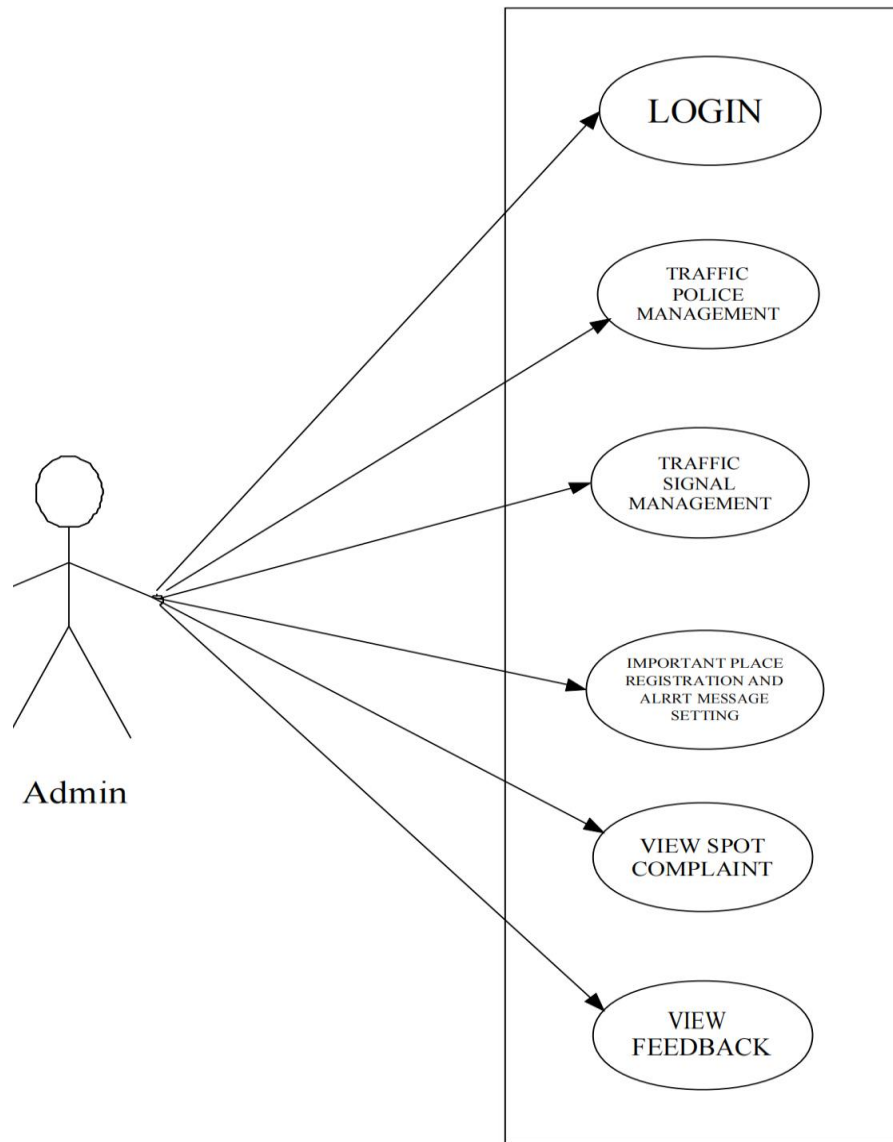
User can view traffic signals.

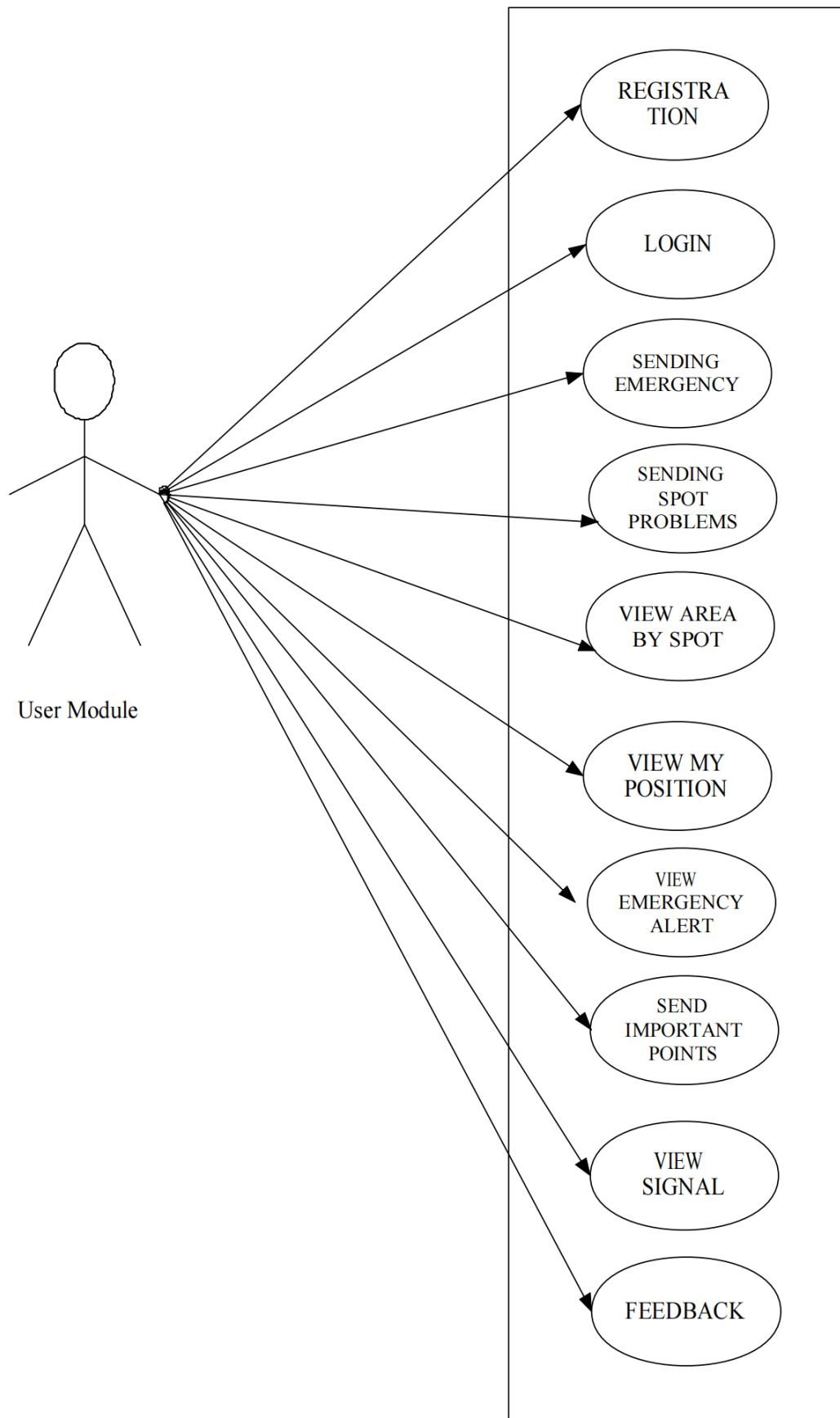
**Feedback:**

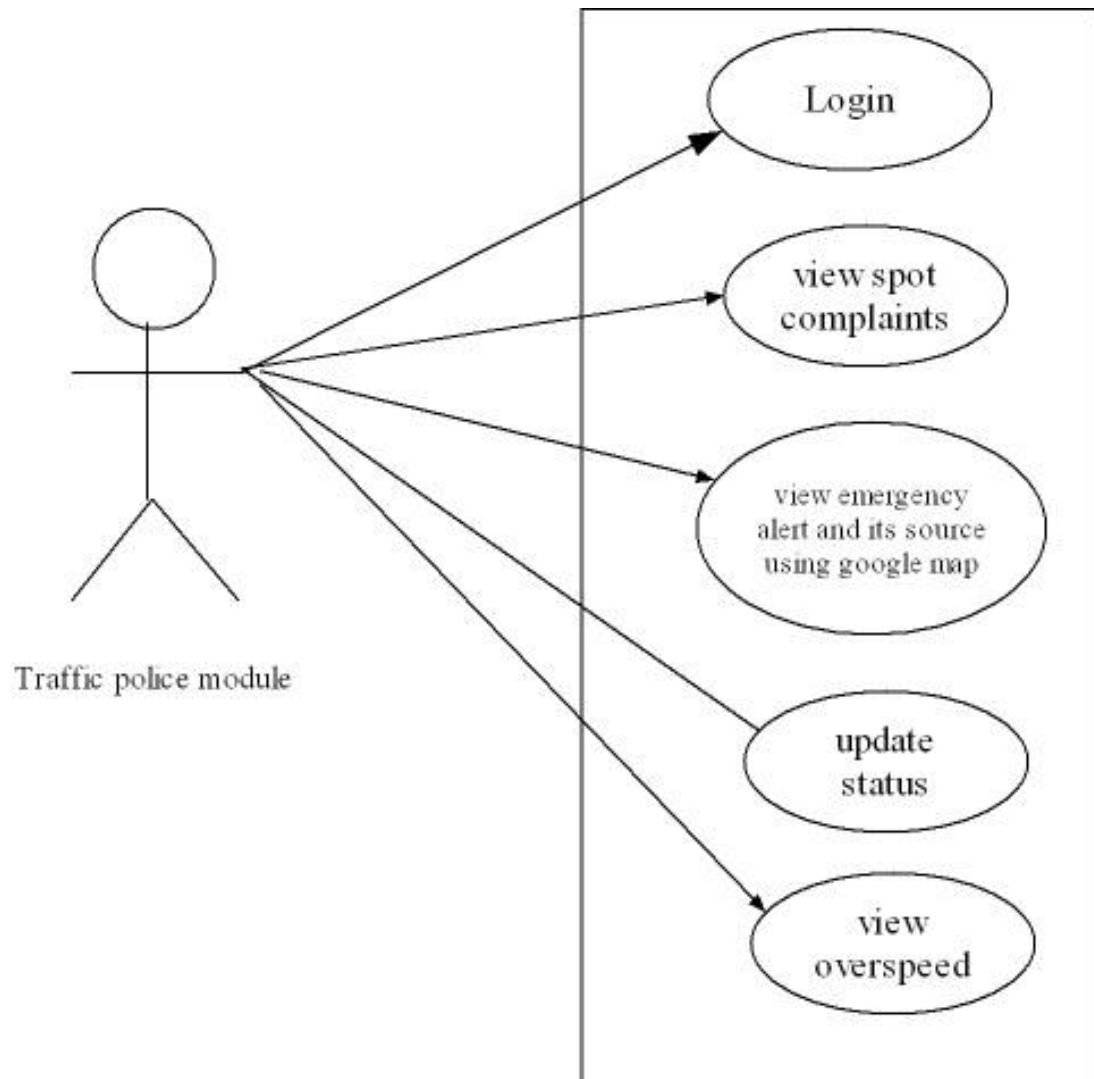
User can give feedback on there experience.



### 2.6.7 USE CASE DIAGRAM







### **3. SYSTEM DESIGN**

### **3.1 INTRODUCTION:**

System design provides an understanding of the procedural details, necessary implementing the system recommended in the feasibility study. Basically it is all about the creation of a new system. This is a critical phase since it decides the quality of the system and has a major impact on the testing and implementation phases.

**System design consists of three major steps.**

- Drawing of the expanded system data flow charts to identify all the processing functions required.
- The allocation of the equipment and the software to be used.
- The identification of the test requirements for the system.

### **CHARACTERS OF DESIGN**

- A design should exhibit a hierarchical organization that makes intelligent use of control among components of the software.
- A design should be modular that is, the software should be logical.
- A design should contain distinct and separable representation of data and procedure.
- A design should lead to interface that reduce the complexity of the connections between modules and with the external environment.

### **3.2 Database Design**

A Database is a collection of inter related data stored with minimum redundancy to serve many users quickly and efficiently. In database design data independence, accuracy, privacy and security are given higher priority. Database design is an integrated approach to the file design. This activity deals with the design of the physical data base. All entities and attributes have been identified while creating the database. The database design deals with the grouping of data into number of tables so as to.

- ✓ Reduplication of data.
- ✓ Minimize storage space.
- ✓ Retrieve the data efficiently.

Following are some guidelines for the database design:

- Design a relational schema so that it is easy to explain its meaning. Do not combine attributes from multiple entity and relationship type into a single relation.
- Design the database schema so that no insertion, deletion or modification anomalies are present in the relation.
- As far as possible, avoid placing attributes in the base relation whose values may frequently be null.
- Design relation schema so that they can be joined with equality conditions on attributes that are either primary keys or foreign keys in a way that no spurious tuples are generated.

### **3.3 Table Design**

DB design is required to manage large bodies of information. The management of data involves both the definition of the structure of storage of information and provisions of mechanism for the manipulation of information. For developing an efficient database certain conditions have to be fulfilled such as:

- Control Redundancy
- Ease of Use
- Data Independence
- Accuracy and Integrity

There are five major steps in design process:

- Identify the table and relationship.
- Identify the data that is needed for each table and relationship.
- Resolve the relationship.
- Verify the design.
- Implement the design

The Database Consist of the following tables given below.

### *1.Login table*

Colum name	Data type	Constraints	Description
login_id	int	primary key	unique identifier
username	varchar(50)	not null	name of user
password	varchar(50)	not null	secret key
type	varchar(50)	not null	To specify the role

### *2.Emergency alert table*

Colum name	Data type	Constraints	Description
Alert_id	Int(15)	primary key	unique identifier
User_id	Int(11)	not null	unique identifier
Date	varchar(15)	not null	Specify date
Time	Varchar(15)	not null	Specify time
Alert	Varchar(50)	Not null	Show alert
Latitude	Varchar(50)	not null	Specify latitude
Longitude	Varchar(50)	not null	Specify longitude
Status	Varchar(50)	not null	Specify status

### *3.Complaint table*

Colum name	Data type	Constraints	Description
Complaint_id	int	primary key	unique identifier
User_id	int	not null	
latitude	varchar(50)	not null	
longitude	varchar(50)	not null	
image	Varchar(200)	not null	
content	varchar(200)	not null	
date	varchar(50)	not null	
time	varchar(50)	not null	

*4.Important\_place table*

Colum name	Data type	Constraints	Description
Place_id	int	primary key	unique identifier
Place_name	varchar	not null	
district	varchar	not null	
Place_type	varchar	not null	
description	varchar	not null	
picture	varchar	not null	
latitude	varchar	not null	
longitude	varchar	not null	

*5.Traffic signal table*

Colum name	Data type	Constraints	Description
Signal_id	int	primary key	unique identifier
place	varchar(200)	foreign key	
district	varchar	not null	Date of complaint
latitude	varchar	not null	
longitude	varchar	not null	Date of replay

*6.Points table*

Colum name	Data type	Constraints	Description
Point_id	int	primary key	
Place_id	int	not null	
Police_id	int	not null	
date	Varchar	not null	
time	varchar	not null	
tittle	Varchar	not null	
content	Varchar	not null	



*7. Feedback table*

Colum name	Data type	Constraints	Description
Feedback_id	int	foreign key	login identifier
User_id	int	not null	
feedback	varchar	not null	
date	varchar	not null	

*8. Location table*

Colum name	Data type	Constraints	Description
Location_id	int	primary key	
User_id	int	not null	
latitude	varchar	not null	
longitude	varchar	Not null	

*9. Overspeed table*

Colum name	Data type	Constraints	Description
Overspeed_id	int	Not null	
User_id	int	not null	
date	varchar	Not null	
time	varchar	Not null	
speed	varchar	Not null	

*10. Road\_condition table*

Colum name	Data type	Constraints	Description
Road_id	int	Primary key	login identifier
date	Varchar(15)	Not null	
time	Varchar(15)	Not null	
latitude	varchar(50)	Not null	
longitude	varchar(50)	Not null	
condition	varchar(50)	Not null	

*11. Traffic police table*

Colum name	Data type	Constraints	Description
Police_id	int	Primary key	
name	varchar	Not null	
Email	varchar	Not null	
Phone	varchar	Not null	
Police_station	varchar	Not null	
Image	varchar	Not null	
Latitude	varchar	Not null	
Longitude	varchar	Not null	

*12. User table*

Colum name	Data type	Constraints	Description
User_id	int	Primary key	
name	varchar	Not null	
email	varchar	Not null	
phone	varchar	Not null	
house	varchar	Not null	
place	varchar	Not null	
post	varchar	Not null	
pin	varchar	Not null	

### 3.5. Data Flow Diagram

A graphical representation is used to describe and analyses the movement of data through a system manual or automated including the processes, Storing of data and delays in the system. Data flow diagrams are the central tool and the basis from which other components are developed.

The transformation of data, from input to output through process may be described logically and independently of the physical components associated with the system.

They are termed logical dataflow diagrams, showing the actual implementations and the movement of data between people, departments and workstations. DFD is one of the most important modelling tools used in system design. DFD shows the flow of data through different process in the system.

#### **PURPOSE:**

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have reasonably completed model of the behavior of the system. It is important to avoid premature designs, wherein develop designs before analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

Throughout my project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In my opinion, "efficient design of the data flow and context flow diagram helps to design the system successfully without much major flaws within the scheduled time". This is the most complicated part in a project. In the designing process, my project took more than the activities in the software lifecycle. If we design a system efficiently with all the future enhancements the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as a way of expressing system requirements in graphical form. A data flow diagram also known as "bubble chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It functionality decomposes the requirement specification down to the lowest level. Data Flow Diagram depicts the information flow, the transformation flow and the transformations that are applied as data move from input to output. Thus DFD describes what data flows rather than how they are processed.

Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is one of the most important tools used during system analysis. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

Data Flow Diagrams are made up of a number of symbols, which represents system components. Data flow modelling method uses four kinds of symbols, which are used to represent four kinds of system components.

These are

- Process
- Data stores
- Data flows
- External entity

**Process:**

Process shows the work of the system. Each process has one or more data inputs and produce one or more data outputs. Processes are represented by rounded rectangles in Data Flow Diagram. Each process has a unique name and number. This name and number appears inside the rectangle that represents the process in a Data Flow Diagram.

**Data Stores:**

A data store is a repository of data. Processes can enter data into a store or retrieve the data from the data store. Each data store has a unique name.

**Data Flows:**

Data flows show the passage of data in the system and are represented by lines joining system components. An arrow indicates the direction of flow and the line is labelled by name of the dataflow.

**External Entity:**

External entities are outside the system but they either supply input data into the system or use other systems output. They are entities on which the designer has control. They may be an organization's customer or other bodies with which the system interacts. External entities that supply data into the system are sometimes called sources. External entities that use the system data are sometimes called sinks. These are represented by rectangles in the

Data Flow Diagram.

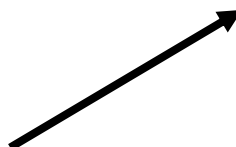
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Basic data flow diagram symbols are.....

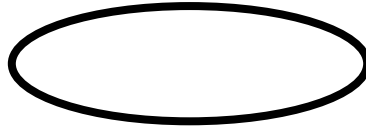
- A Square defines a source (originator) or destination of a system data:



- An Arrow identifies data flow. It is a pipeline through which information flows:



- A Circle represents a process that transforms incoming data flow(s) into outgoing data flow(s):



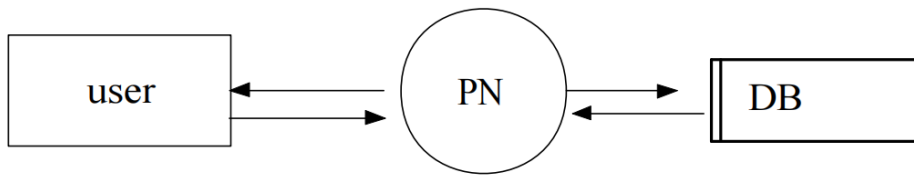
- An Open Rectangle is a data store:



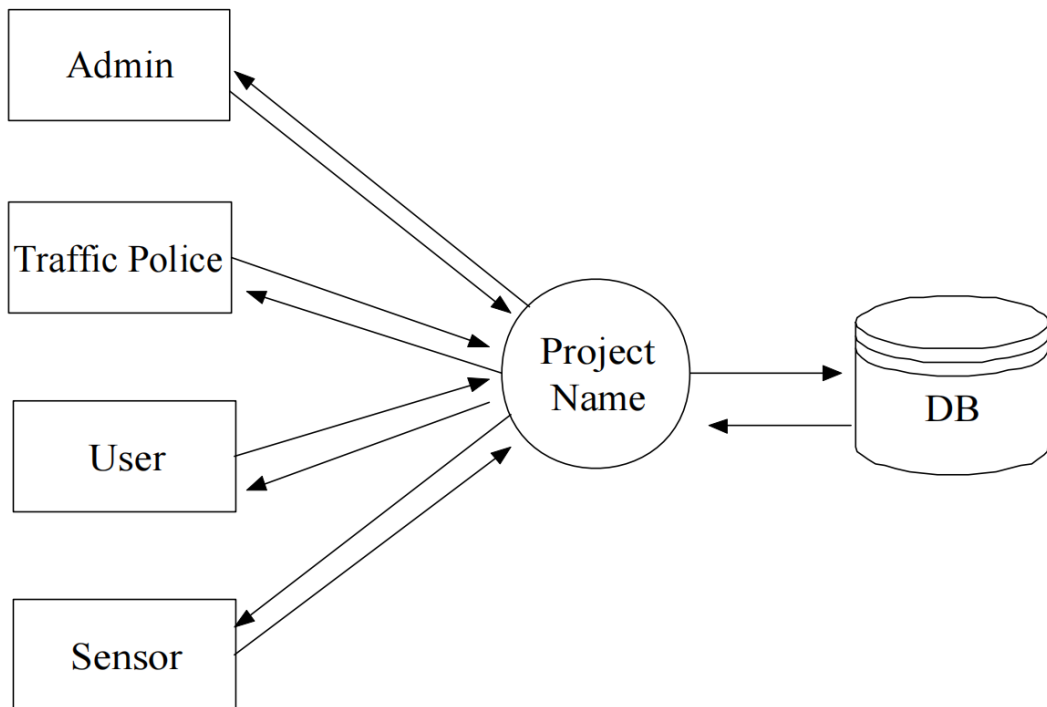
**Four steps are commonly used to construct a DFD:**

- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and left to right.
- When a process is exploded into lower level details they are numbered.
- The names of data stores, sources and destinations are written in Capital letters.

### LEVEL- 0

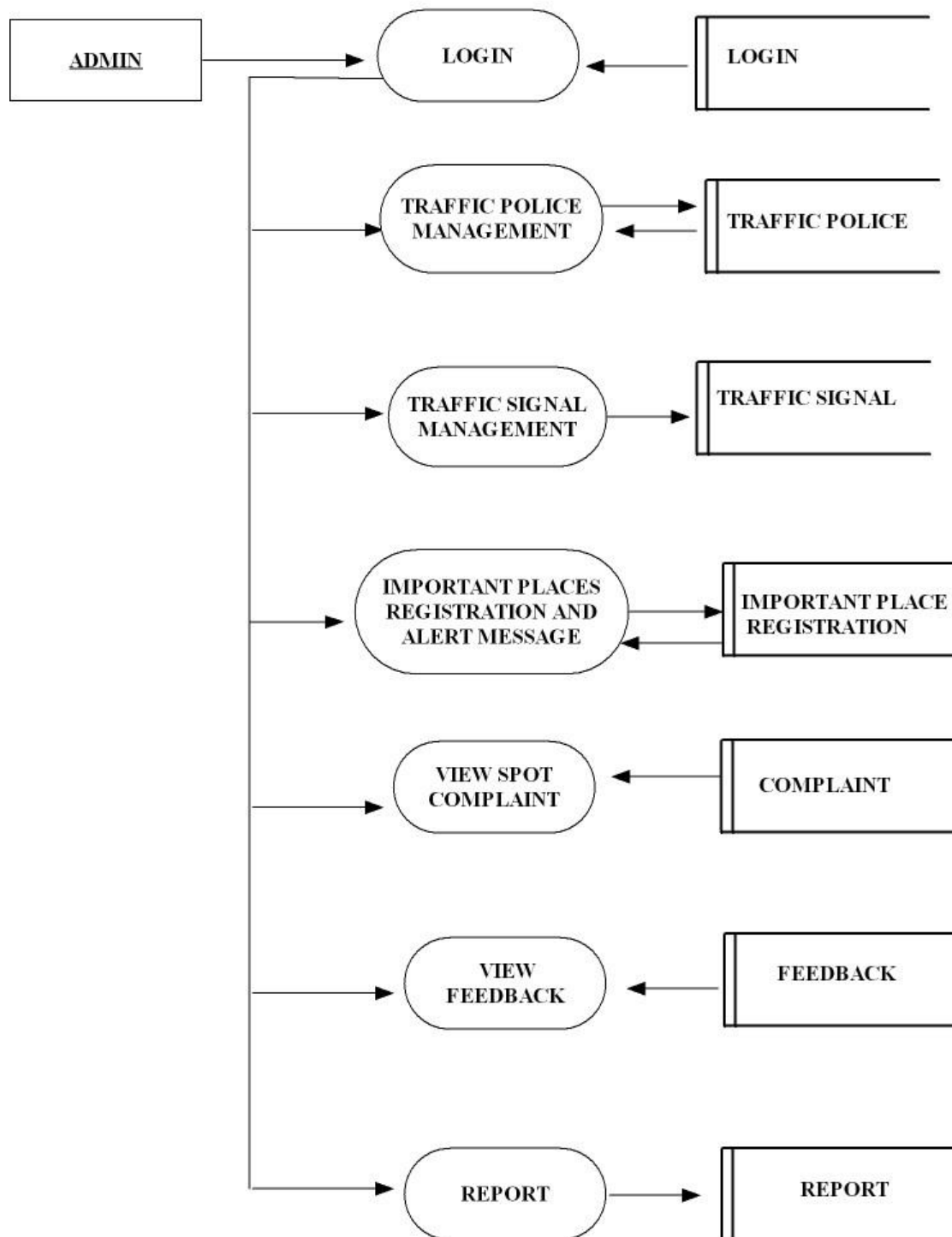


### LEVEL-1



## ADMIN MODULE

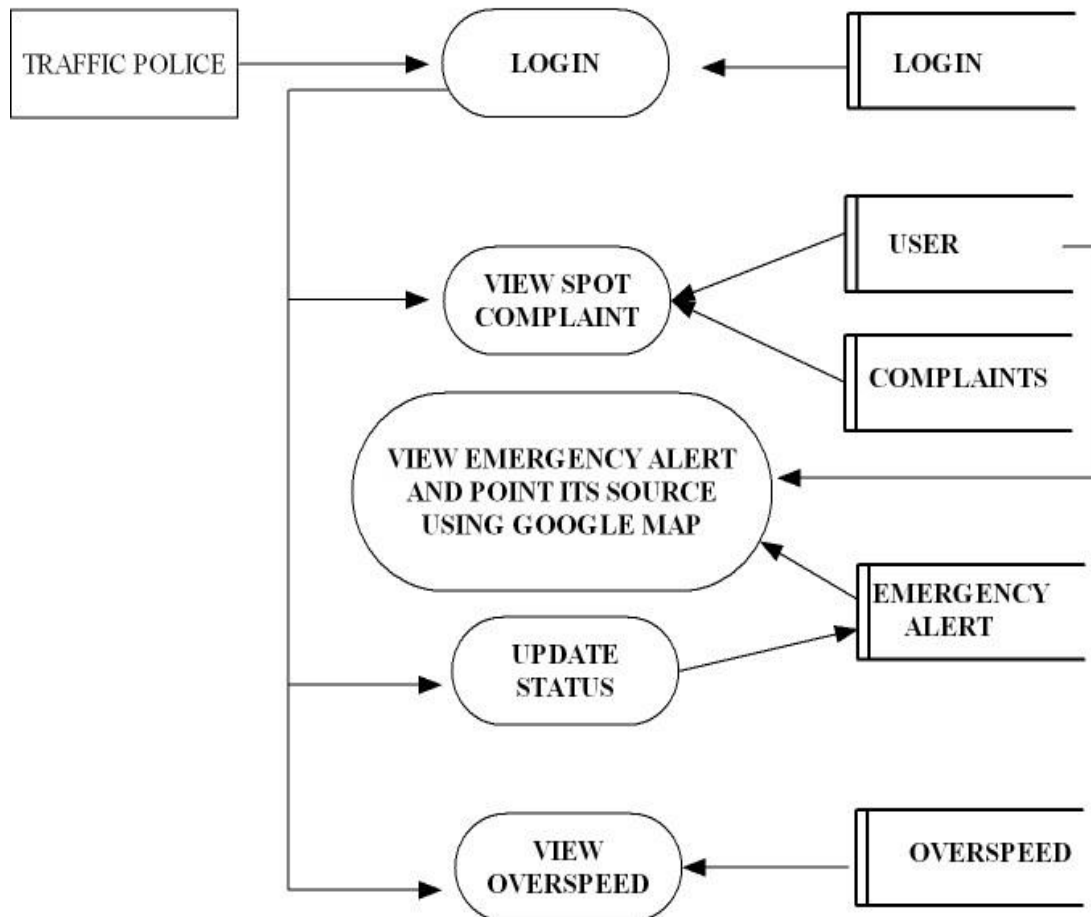
### LEVEL 1-2





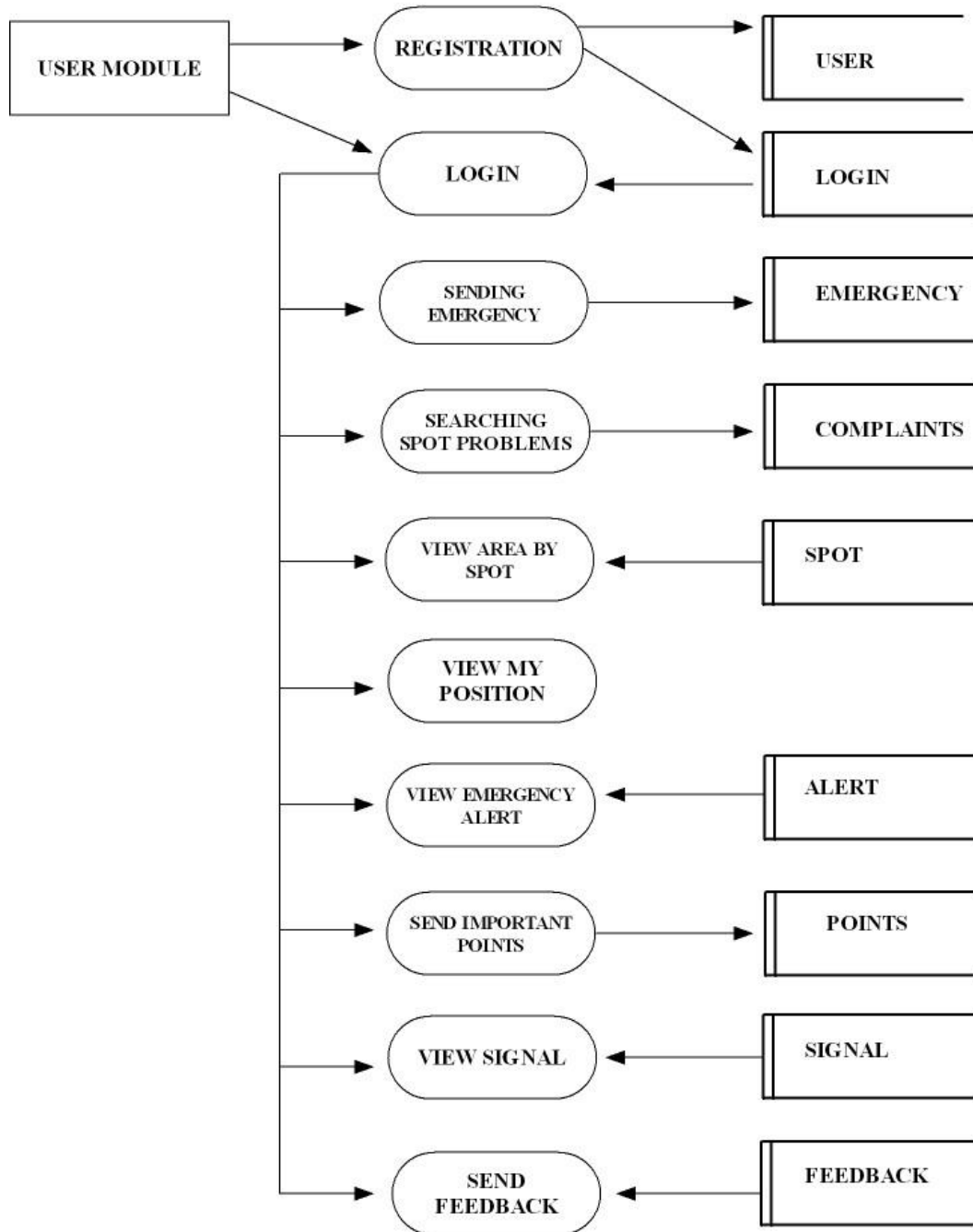
## TRAFFIC POLICE MODULE

### LEVEL 1-3



## USER MODULE

### LEVEL 1-4



### 3.6 ER Diagram

An ER diagram is a diagram that helps to design databases in an efficient way. It is a data model for describing the data or information. It is a visual representation of data that describes how data is related to each other. The main components of ER models are entities, attributes and the relationships that can exist among them.

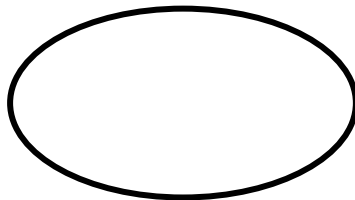
#### Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.



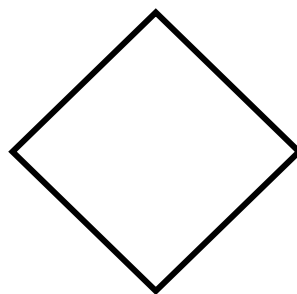
#### Attribute

Attributes are properties of entities. Attributes are represented by means of ellipses. Every ellipse represents one attribute and is directly connected to its entity (rectangle).

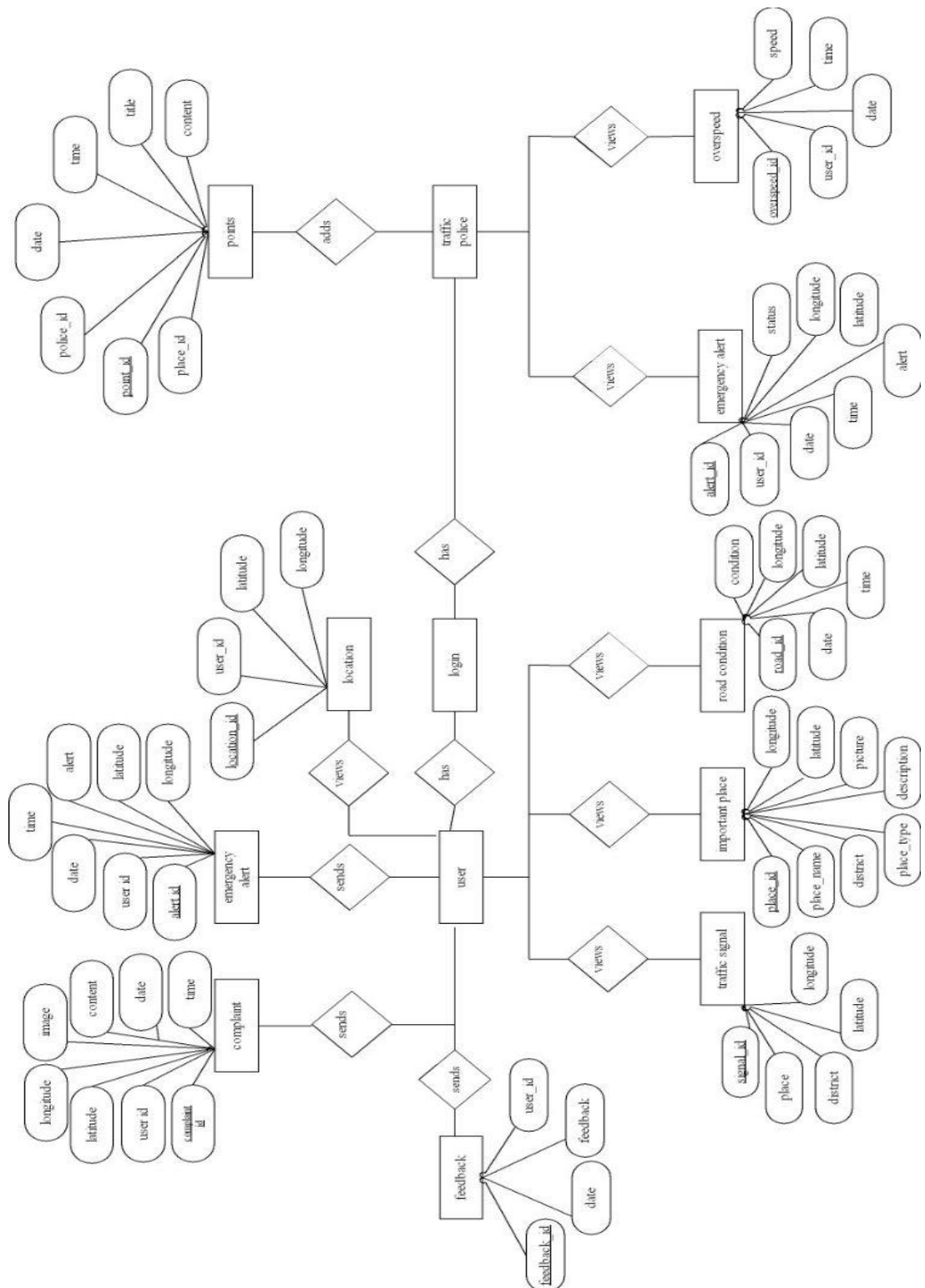


#### Relationship

Relationships are represented by diamond shaped box. Name of the relationship is written in the diamond box. All entities (rectangles), participating in relationship, are connected to it by a line.



## Achitectural design:



## **4.CODING**

## **4.1 INPUT INTERFACE**

Input design is a part of overall system design, which requires very careful attention. If data going into the system is correct, then the processing and output will magnify these errors. Thus the designer has a number of clear objectives in the different stages of input design.

- To produce a cost effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that input is acceptable to and understand by the user.

## **4.2 OUTPUT INTERFACE**

At the beginning of the output design various types of outputs such as external, internal, operational and interactive and turn around are defined. Then the format, content, location, frequency, volume and sequence of the outputs are specified. The content of the output must be defined in detail. The system analysis has two specific objectives at this stage.

- To interpret and communicate the results of the computer part of a system to the users in a form, which they can understand, and which meets their requirements.
- To communicate the output design specifications to programmers in a way in which it is unambiguous, comprehensive and capable of being translated into a programming language.

## **4.3 SOFTWARE DESCRIPTION**

### **4.3.1 HTML**

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML0000 describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML

specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are

having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

HTML files are written in ASCII text, so the user can use any text editor to create his/her web page, though a browser of one sort or another is necessary to view the web page. HTML is case insensitive with its language commands. The characters within the document, however, are case sensitive. The language consists of various "tags" which are known as elements. These allow the browser to understand (and put into the desired/specified format) the layout, background, headings, titles, lists, text and/or graphics on the page. The elements are classified according to their function in the HTML document. There are head elements and body elements. The head elements identify properties of the entire document, while body elements actually mark text as content and show a change in the appearance in one way or another. Most elements have a beginning and an ending which encompass the text the user wishes to mark with the tag. All HTML documents must begin with the element and end with the element. Some of the other elements which may be used are tags to create lists-- both ordered lists as well as unordered lists. The user may also create larger or smaller, bolder, italicized, or underlined text. Attributes may be used along with the elements. These perform functions such as placement of text, indication of the source files of images, and identification of links to the document or part of the document.

### **4.3.2 CSS**

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages, user interfaces for web applications, and user interfaces for many mobile applications. CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colours, and fonts.

#### **Advantages of CSS**

- CSS saves time – you can write CSS once and then reuse same sheet in multiple HTML pages.

You can define a style for each HTML element and apply it to as many Web pages as you want.

- Pages load faster – If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
- Easy maintenance – To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- Superior styles to HTML – CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- Multiple Device Compatibility – Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- Global web standards – Now HTML attributes are being deprecated and it is being recommended to use CSS. So it is a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.
- Offline Browsing – CSS can store web applications locally with the help of an offline cache. Using of this, we can view offline websites. The cache also ensures faster loading and better overall performance of the website.
- Platform Independence – The Script offer consistent platform independence and can support latest browsers as well.

#### **4.4.3 JAVASCRIPT**

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript.

The General-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

Advantages of JavaScript:



- Less server interaction – you can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.
- Immediate feedback to the visitors – They don't have to wait for a page reload to see if they have forgotten to enter something.
- Increased interactivity – you can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.
- Richer interfaces – you can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

#### **4.3.4 MySQL**

MySQL is an open-source relational database management system (RDBMS). MySQL is released under an open-source license. So you have nothing to pay to use it. MySQL is a very powerful program in its own right.

It handles a large subset of the functionality of the most expensive and powerful database packages. It uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.

It works very quickly and works well even with large data sets. It is very friendly to PHP, the most appreciated language for web development. It supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB). It is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

#### **Major features as available in MySQL 5.6**

- A broad subset of ANSI SQL 99, as well as extensions.
- Cross-platform support.
- Stored procedures, using a procedural language that closely adheres to SQL/PSM.
- Triggers.
- Cursors.
- Updatable views.
- Online DDL when using the InnoDB Storage Engine.
- Information schema.

- Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
- A set of SQL Mode options to control runtime behaviour, including a strict mode to better adhere to SQL standards.
- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine.
- Transactions with savepoints when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
- ACID compliance when using InnoDB and NDB Cluster Storage Engines.
- SSL support → Query caching → Sub-SELECTs (i.e. nested SELECTs) .
- Built-in replication support (i.e., master-master replication and master-slave replication) with one master per slave, many slaves per master.
- Multi-master replication is provided in MySQL Cluster, and multimaster support can be added to unclustered configurations using Galera Cluster.
- Full-text indexing and searching.
- Embedded database library.
- Unicode support.
- Partitioned tables with pruning of partitions in optimizer.
- Shared-nothing clustering through MySQL Cluster.
- Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.
- Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.
- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

## **Advantages**

MySQL database server has lots of advantages over its competitors. Some of these advantages have been explained below.

- Open Source and Cost Effective:

The best thing about MySQL server is that this is open source and it has a free version as well. By open source software, we mean that the code of the software is

available and anyone can tailor it according to his requirement. Companies prefer MySQL because they don't have to pay anything for this excellent product.

➤ Portability:

MySQL is cross platform database server. MySQL can be run on a variety of platforms including Windows, OS2, Linux and Solaris. Portability of MySQL server makes it suitable for applications that target multiple platforms particularly web application. MySQL contains API for almost all the major programming languages and can be easily integrated with the languages like PHP, C++, Perl, C, Python and ruby. In fact, MySQL is a part of the famous LAMP (Linux Apache MySQL PHP) server stack which is used worldwide for web application development.

➤ Seamless Connectivity:

Various secure and seamless connection mechanisms are available in order to connect with MySQL server. These connections include named pipes, TCP/IP sockets and UNIX Sockets.

➤ Rapid Development and Continuous Updates:

Being an open source product, MySQL has a very large developer community which releases regular patches and updates for MySQL. Several database templates have been developed which can be readily used and modified resulting in rapid application development.

➤ Security:

MySQL server databases are extremely secure and all the data access scenarios are protected via password and good thing about these passwords is that they are stored in encrypted form and it is not easy to break these advanced and complex encryption algorithms.

### **4.3.5 Python**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code

reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

### Major features of python

- Easy to code
- Free and Open Source
- Object-Oriented Language
- GUI Programming Support
- High-Level Language
- Extensible feature
- Python is **Portable** language
- Python is Integrated language

### Advantages

- Extensive support libraries
- Integration feature

- Improved productivity
- Easy to learn and write
- Vast library support
- Free and open source

#### **4.3.6 Flask**

Flask is a web application framework written in Python. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Pocco. Flask is based on the Werkzeug WSGI toolkit and the Jinja2 template engine. Both are Pocco projects.

It is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier. It gives developers flexibility and is a more accessible framework for new developers since you can build a web application quickly using only a single Python file. Flask is also extensible and doesn't force a particular directory structure or require complicated boilerplate code before getting started.

## **TECHNOLOGY SPECIFICATION**

The proposed system is developed using HTML , CSS, JavaScript for web and Java(Android Studio) for Android as Front end. MySQL ,flask and python for web and Python ,MySQL for Android as Back end.

### **1. FRONT END**

#### **HTML**

HTML is an acronym which stands for Hyper Text Markup Language which is used for creating web pages and web applications. Let's see what is meant by Hypertext Markup Language, and Web page. Hyper Text: HyperText simply means "Text within Text." A text has a link within it, is a hypertext. Whenever you click on a link which brings you to a new webpage, you have clicked on a hypertext. HyperText is a way to link two or more web pages (HTML documents) with each other. Markup language: A markup language is a computer language that is used to apply layout and formatting conventions to a text document. Markup language makes text more interactive and dynamic. It can turn text into images, tables, links,

etc. Web Page: A web page is a document which is commonly written in HTML and translated by a web browser. A web page can be identified by entering an URL. A Web page can be of the static or dynamic type. With the help of HTML only, we can create static web pages. Hence, HTML is a markup language which is used for creating attractive web pages with the help of styling, and which looks in a nice format on a web browser. An HTML document is made of many HTML tags and each HTML tag contains different content

## **Features**

- 1) It is a very easy and simple language. It can be easily understood and modified.
- 2) It is very easy to make an effective presentation with HTML because it has a lot of formatting tags.
- 3) It is a markup language, so it provides a flexible way to design web pages along with the text.
- 4) It facilitates programmers to add a link on the web pages (by html anchor tag), so it enhances the interest of browsing of the user.
- 5) It is platform-independent because it can be displayed on any platform like Windows, Linux, and Macintosh, etc.
- 6) It facilitates the programmer to add Graphics, Videos, and Sound to the web pages which makes it more attractive and interactive.
- 7) HTML is a case-insensitive language, which means we can use tags either in lowercase or upper-case.

## **HTML Versions**

Since the time HTML was invented there are lots of HTML versions in market, the brief introduction about the HTML version is given below:

**HTML 1.0:** The first version of HTML was 1.0, which was the barebones version of HTML language, and it was released in 1991.

**HTML 2.0:** This was the next version which was released in 1995, and it was standard language version for website design. HTML 2.0 was able to support extra features such as form-based file upload, form elements such as text box, option button, etc.

**HTML 3.2:** HTML 3.2 version was published by W3C in early 1997. This version was capable of creating tables and providing support for extra options for form elements. It can also support a web page with complex mathematical equations. It became an official

standard for any browser till January 1997. Today it is practically supported by most of the browsers.

**HTML 4.01:** HTML 4.01 version was released on December 1999, and it is a very stable version of HTML language. This version is the current official standard, and it provides added support for stylesheets (CSS) and scripting ability for various multimedia elements.

**HTML5 :** HTML5 is the newest version of Hyper Text Markup language. The first draft of this version was announced in January 2008. There are two major organizations one is W3C (World Wide Web Consortium), and another one is WHATWG( Web Hypertext Application Technology Working Group) which are involved in the development of HTML 5 version, and still, it is under development.

## Description of HTML example

**<!DOCTYPE>:** It defines the document type or it instruct the browser about the version of HTML

**<html> :** This tag informs the browser that it is an HTML document. Text between html tag describes the web document. It is a container for all other elements of HTML except<!DOCTYPE>

**<head> :** It should be the first element inside the element, which contains the metadata(information about the document). It must be closed before the body tag opens.

**<title>:** As its name suggests it is used to add title of that html page which appears at the top of the browser window. It must be placed inside the head tag and should close immediately (optional).

**<body> :** Text between body tag describes the body content of the page that is visible to the end user . This tag contains the main content of the html document.

**<h1> :** Text between <h1> tag describes the first level heading of the webpage.

**<p>:** Text between <p> tag describes the paragraph of the webpage.

## **CSS (Cascading Style Sheet)**

CSS is used to control the style of a web document in a simple and easy way. CSS is the acronym for "**Cascading Style Sheet**". **Cascading Style Sheets (CSS)** is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript .

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

### **History of CSS**

CSS was first proposed by **Hakon Wium Lie** on October 10, 1994. At the time, Lie was working with Tim Berners-Lee (father of Html) at CERN. The European Organization for Nuclear Research is known as CERN. Hakonwium lie is known as father of css. CSS was proposed in 1994 as a web styling language, to solve some of the problems of Html 4. There were other styling languages proposed at this time, such as Style Sheets for Html and JSSS but CSS won.

### **Why to learn CSS?**

**Cascading Style Sheets**, fondly referred to as **CSS**, is a simple design language intended to simplify the process of making web pages presentable.<sup>23</sup> CSS is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain.

I will list down some of the key advantages of learning CSS:

- **Create Stunning Web site** - CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.



- **Become a web designer** - If you want to start a carrier as a professional web designer, HTML and CSS designing is a must skill.
- **Control web** - CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.
- **Learn other languages** - Once you understand the basic of HTML and CSS then other related technologies like javascript, php, or angular are become easier to understand.

## Types of CSS

There are three ways of inserting a style sheet in any Html documents, they are given below:

- Inline style sheet
- Internal style sheet
- External style sheet

Inline CSS is use with any elements of HTML where it is used on page. Here we use inline CSS for paragraph, the example shows how to change the color and the left margin of a paragraph. An internal style sheet should be used when a single document has a unique style. Internal styles sheet is defined in the head section of an HTML page, by using the <style> tag. An external style sheet is ideal when the style is applied to many pages. With an external style sheet, we can change the look of an entire Web site by changing one file.

## CSS Selectors

Selectors are used for select an Html element it is select by name, id, class etc.

1. id selector
2. class selector
3. Element Selector
4. Group Selector
5. Universal Selector

## Features

1. A style rule consists of a selector component and a declaration block component.
2. The selector is used to point to the HTML component which you want to get styled.

3. Inside the declaration block, one or more declarations are contained along with semicolons.
4. Every declaration which is put has a CSS property name, a semicolon, and a value. For example, color is the property and the value is red in color. Font size is the property and the 15px is the value.
5. CSS declaration ends with a semicolon and these blocks are surrounded by curly braces.
6. CSS selectors are the ones which are used to find HTML elements which are based on the element name, id, attribute, class and more.
7. One unique element will be selected by the ID of an element.
8. If you wish to select the particular element with a specific id, the # function along with the id attribute should be used.
9. If you wish to select the elements with a specific class, the period character along with the name class should be written.
10. Universal selector: If you are not interested in choosing the elements of a certain type, the universal selector simply matches with the element name.
11. Element selector: These selectors choose the element based on the element name.
12. Descendent selector: When a particular element lies inside another element, then it is called as the descendent selector.
13. ID selector: This selector uses the id of the HTML element so that a specific element could be selected.
14. Class selectors: It selects the element with a specific class attribute.
15. Grouping selectors: It will be a good option to group the selectors so as to minimize the code. Each selector along with a comma should be used to group the selectors.

## JavaScript

**JavaScript** is a object-based scripting language and it is light weighted. It is first implemented by Netscape (with help from Sun Microsystems). JavaScript was created by Brendan Eich at Netscape in 1995 for the purpose of allowing code in web-pages (performing logical operation on client side).It is not compiled but translated. JavaScript Translator is responsible to translate the JavaScript code which is embedded in browser.Netscape first introduced a JavaScript interpreter in Navigator 2. The interpreter was

25 an extra software component in the browser that was capable of interpreting JavaScript source code inside an HTML document. This means that web page developers no need other software other than a text editor of develop any web page.

## **Why we use JavaScript?**

Using HTML we can only design a web page but you cannot run any logic on web browser like addition of two numbers, check any condition, looping statements (for, while), decision making statement (if-else) at client side. All these are not possible using HTML so for perform all these task at client side you need to use JavaScript

## **History**

JavaScript is an object-based scripting language and it is light weighted. It is first implemented by Netscape (with help from Sun Microsystems). JavaScript was created by Brendan Eich at Netscape in 1995 for the purpose of allowing code in web-pages (performing logical operation on client side). Using HTML we can only design a web page but you cannot run any logic on web browser like addition of two numbers, check any condition, looping statements (for, while), decision making statement (if-else) etc. All these are not possible using HTML so to perform all these task, we use JavaScript. Using HTML we can only design a web page if we want to run any programs like c programming we use JavaScript. Suppose we want to print sum of two numbers then we use JavaScript for coding.

## **Features**

- JavaScript is an object-based scripting language.
- Giving the user more control over the browser.
- It Handling dates and time.
- It Detecting the user's browser and OS
- It is light weighted.
- JavaScript is a scripting language and it is not java.
- JavaScript is interpreter based scripting language.
- JavaScript is case sensitive.
- JavaScript is object based language as it provides predefined objects.

- Every statement in JavaScript must be terminated with semicolon (;).
- Most of the JavaScript control statements syntax is same as syntax of control statements in C language. An important part of JavaScript is the ability to create new functions within scripts. Declare a function in JavaScript using function keyword

## Android

**Android** is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language. Android applications are written in the Java programming language. The Android SDK tools compile the code—along with any data and resource files— into an Android package, an archive file with an .apk suffix. All the code in a single .apk file is considered to be one application and is the file that Android-powered devices use to install the application. Application components are the essential building blocks of an Android application. Each component is a different point through which the system can enter your application. Not all components are actual entry points for the user and some depend on each other, but each one exists as its own entity and plays a specific role—each one is a unique building block that helps define application's overall behavior.

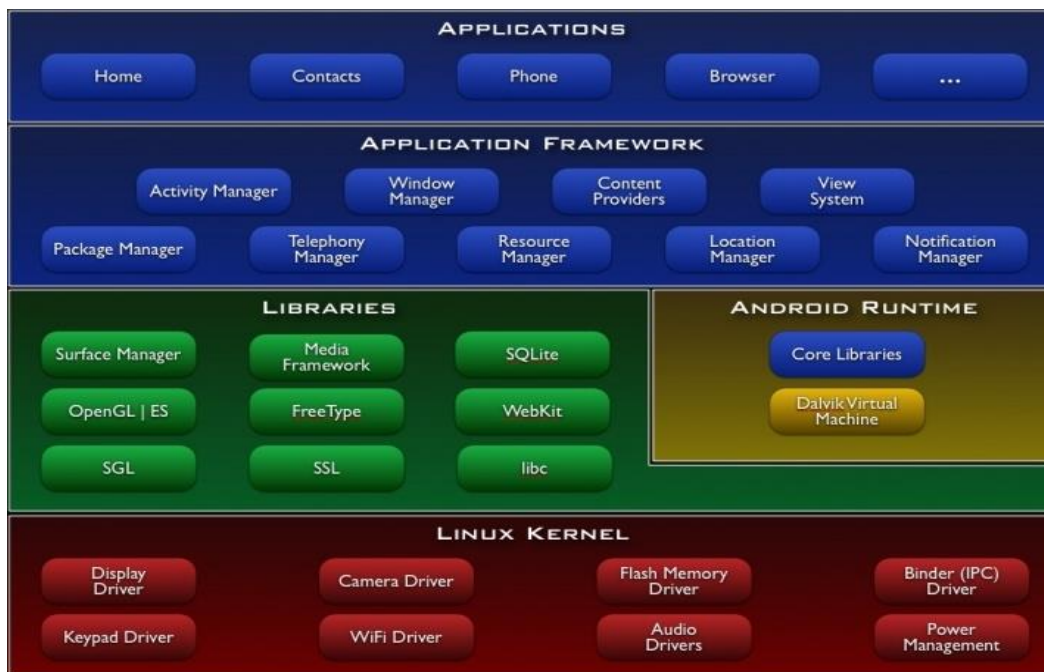
## Features

- Application framework enabling reuse and replacement of components
- Dalvik virtual machine optimized for mobile devices
- Integrated browser based on the open source Web Kit engine
- Optimized graphics powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
- Media support for common audio, video, and still image formats (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
- GSM Telephony (hardware dependent)

- Bluetooth, EDGE, 3G, and Wi-Fi (hardware dependent)
- Camera, GPS, compass, and accelerometer (hardware dependent)
- Rich development environment including a device emulator, tools for debugging, memory and performance profiling, and a plug-in for the Eclipse IDE.
- 

## ANDROID ARCHITECTURE

The following diagram shows the major components of the Android operating system. Each section is described in more detail below.



## APPLICATION FRAMEWORK

By providing an open development platform, Android offers developers the ability to build extremely rich and innovative applications. Developers are free to take advantage of the device hardware, access location information, run background services, set alarms, add notifications to the status bar, and much, much more. Developers have full access to the same framework APIs used by the core applications. The application architecture is designed to simplify the reuse of components; any application can publish its capabilities and any other application may then make use of those capabilities (subject to security constraints enforced

by the framework). This same mechanism allows components to be replaced by the user. Underlying all applications is a set of services and systems, including:

A rich and extensible set of the views that can be used to build an application, including lists, grids, text boxes, buttons, and even an embeddable web browser

Content Providers that enable applications to access data from other applications (such as Contacts), or to share their own data A Resource Manager, providing access to non-code resources such as localized strings, graphics, and layout files

A Notification Manager that enables all applications to display custom alerts in the status bar

An Activity Manager that manages the lifecycle of applications and provides a common navigation back stack.

## **Libraries**

Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below:

- System C library - a BSD-derived implementation of the standard C system library (libc), tuned for embedded Linux-based devices
- Media Libraries - based on Packet Video's Open CORE; the libraries support playback and recording of many popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG
- Surface Manager - manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications
- LibWebCore - a modern web browser engine which powers both the Android browser and an embeddable web view
- SGL - the underlying 2D graphics engine
- 3D libraries - an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer
- Free Type - bitmap and vector font rendering<sup>24</sup>

## Android Runtime

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language. Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple VMs efficiently. The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been transformed into the .dex format by the included "dx" tool. The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management. Linux Kernel Android relies on Linux version 2.6 for core system services such as security, memory management, process management, network stack, and driver model. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack.

## Activity Lifecycle

Activities in the system are managed as an activity stack. When a new activity is started, it is placed on the top of the stack and becomes the running activity -- the previous activity always remains below it in the stack, and will not come to the foreground again until the new activity exits.

An activity has essentially four states:

- If an activity in the foreground of the screen (at the top of the stack), it is active or running.
- If an activity has lost focus but is still visible (that is, a new non-full sized or transparent activity has focus on top of your activity), it is paused. A paused activity is completely alive (it maintains all state and member information and remains attached to the window manager), but can be killed by the system in extreme low memory situations.
- If an activity is completely obscured by another activity, it is stopped. It still retains all state and member information, however, it is no longer visible to the user so its window is hidden and it will often be killed by the system when memory is needed elsewhere.

- If an activity is paused or stopped, the system can drop the activity from memory by either asking it to finish, or simply killing its process. When it is displayed again to the user, it must be completely restarted and restored to its previous state.

## **Android Studio**

**Android Studio** is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (ADT) as primary IDE for native Android application development. Android Studio supports all the same programming languages of IntelliJ, and PyCharm and Android Studio 3.0 supports Java 7 language features and a subset of Java 8 language features that vary by platform version. Features like Gradle-based build support, Android-specific refactoring and quick fixes, a rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations, Android Virtual Device (Emulator) to run and debug apps in the Android studio, etc. are provided in the current stable version.

## **About JAVA**

The first version of Java began in 1991 and was written in 18 months at Sun micro system. In fact, it wasn't even called Java in those days it was Oak, and it was used internally at sun.

Java had adopted a model that made it perfect for the Internet, the byte code model. It is implemented as the Java virtual Machine (JVM), which is the application that actually runs the java program. 26 When JVM is installed on a computer, it can run java programs. Java programs, before ,don't need to be self-sufficient, and they don't have to include all the machine –level code that actually runs on the computer .In this way ,our Java program can be very small, because all the machine-level code to run our program is already on the target computer and doesn't have to be downloaded.

When it executes a program, the JVM can strictly monitor what goes on, which makes it great for Internet Applications.



## **JAVA FEATURES**

The inventors of java wanted to design a language, which could offer solutions to some of the problems encountered in modern programming. They wanted the language to be reliable, portable and distributed but also simple, compact and interactive. Sun Microsystems officially describes java with the following attributes.

- Compiled and Interpreted
- Platform-Independent and Portable
- Object-Oriented
- Robust and Secure
- Distributed
- Familiar, Simple and Small
- Multithreaded and Interactive
- High Performance
- Dynamic and Extensible

### **Compiled and Interpreted**

Usually a computer language is either compiled or interpreted. Java combines both these approaches thus making java a two-stage system first java compiler translate source code in to byte code instructions. Byte-codes are not machine code that can be directly executed by the machine that is running the java program. Platform Independent and Portable.

The most significant contribution java over other languages is its portability. Java programs can be easily moved from one computer system to another, anywhere at any time. Changes and Upgrades in 27 operating systems, processors and system resources will not force any changes in java programs. This is the reason why java has become a popular language for programming on internet, which interconnects different kinds of systems worldwide. Java ensures portability in two ways. First, java compiler generates byte code

instructions that can implemented on any machine. Secondly, the sizes of the primitive data types are machine independent.

## **Object-Oriented**

Java is a true Object-Oriented Language. Almost everything in Java is an Object. All program code and data reside within objects and classes. Java comes with an extensive set of classes arranged in packages that we can use in our programs by inheritance. The object model in java is simple and easy to extend.

## **Robust and Secure**

Java is a robust language. It provides many safe guards to ensure reliable code. It has strict compile time checking for data types. It is designed as garbage collected language. Java also incorporates with the concept of exception handling.

## **Distributed**

Java is designed as a distributed language for creating application on network. It has the ability to share both data & Program.

## **Multithreaded and interactive**

Multithreaded means handling multiple tasks simultaneously java supports multithreaded programs. This means that we not wait for the application to finish one task before beginning another.

## **High performance**

Java performance is impressive for an interpreted language mainly due to the use of intermediate byte code. Java architecture is also designed to reduce overheads during runtime.

## **2 BACK END**

### **Python**

Python is an object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting

or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

## MySQL

MySQL software is Open Source

Open source means that it is possible for anyone to use and modify. Anybody can download the MySQL software from the Internet and use it without paying anything. The MySQL database server is very fast, reliable, and easy to use. It was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Though under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet

- An object-oriented interface
- Support for prepared statements
- Support for multiple statements
- Support for transactions
- Enhanced debugging support

- Embedded server support

## Pycharm:

**PyCharm** is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains (formerly known as IntelliJ).<sup>[5]</sup> It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as data science with Anaconda.<sup>[6]</sup>

### FEATURES

- Coding assistance and analysis, with code completion, syntax and error highlighting, linter integration, and quick fixes
- Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
- Python refactoring: includes rename, extract method, introduce variable, introduce constant, pull up, push down and others
- Integrated Python debugger
- Integrated unit testing, with line-by-line code coverage
- Version control integration: unified user interface for Mercurial, Git, Subversion, Perforce and CVS with change lists and merge

## Flask

**Flask** is a micro web framework written in Python. It is classified as microframework because it does not require particular tools or libraries.<sup>[2]</sup> It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

## **5.CODING PAGES**

## 5.1 Admin Page

```
@app.route("/login_post", methods=['post'])

def login_post():

    username=request.form['textfield']

    pasword=request.form['textfield2']

    obj=Db()

    res=obj.selectOne("select * from login where username='"+username+"' and
password='"+pasword+"'")

    if res is not None:

        if res['type']=='admin':

            session['lg']='lin'

            return redirect('/admin_home')

        elif res['type']=='police':

            session['lid']=res['login_id']

            session['lg']='lin'

            return redirect('/traffic_home')

        else:

            return '<script>alert("invalid");window.location="/admin_home"</script>'

    else:

        return '<script>alert("invalid");window.location="/admin_home"</script>'

@app.route('/admin_home')

def admin_home():

    if session['lg']!='lin':
```

```

        return '<script>alert("you are logged out");window.location="/"<script>'

db=Db()

qry=db.select("select * from feedback,`user` where feedback.user_id=`user`.user_id")

return render_template("admin/admin_index.html",data=qry)

@app.route('/add_device_allocation',methods=['get'])

def add_device_allocation():

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    obj=Db()

    res=obj.select("select * from traffic_police")

    return render_template('admin/device_allocation.html',data=res)

@app.route("/adde_device_allocation_post",methods=['post'])

def add_device_allocation_post():

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    poli=request.form['select']

    Model=request.form['textfield']

    Imei=request.form['textfield2']

    obj=Db()

    obj.insert("insert into device_allocation values('"+str(poli)+"','"+Model+"','"+Imei+"")")

    return'<script>alert("allocated");window.location="/admin_home"</script>'

@app.route('/add_important_place')

def add_important_place():

    if session['lg']!= 'lin':

```

```

        return '<script>alert("you are logged out");window.location="/"<script>'

    return render_template('admin/add_important_places.html')

@app.route('/add_important_place_post',methods=['post'])

def add_important_places():

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    District=request.form['select']

    place_name=request.form['textfield']

    place_type=request.form['select1']

    description=request.form['textfield2']

    image = request.files['fileField']

    lati=request.form['textfield4']

    longi=request.form['textfield5']

    date = datetime.datetime.now().strftime("%y%m%d-%H%M%S")

    image.save(r"C:\Users\anumo\PycharmProjects\Wolverine\static\pic\\" + date + '.jpg')

    p = "/static/pic/" + date + '.jpg'

    obj=Db()

    obj.insert("insert                                     into
important_place(place_name,district,place_type,description,picture,latitude,longitude)values(
"+place_name+"','"+str(District)+"','"+place_type+"','"+description+"','"+str(p)+"','"+lati+"','"
+longi+"')")

    return '<script>alert("added
succesfully");window.location="/admin_home#contact"</script>'

@app.route('/add_police')

def add_police():

```



```

if session['lg']!= 'lin':

    return '<script>alert("you are logged out");window.location="/"<script>'

return render_template('admin/add_police.html')

@app.route('/add_police_post',methods=['post'])

def add_police_post():

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    name=request.form['textfield']

    police_station=request.form['textfield2']

    image=request.files['fileField']

    phone=request.form['textfield3']

    email=request.form['textfield4']

    latitude=request.form['textfield5']

    longitude=request.form['textfield6']

    date=datetime.datetime.now().strftime("%y%m%d-%H%M%S")

    image.save(r"C:\Users\anumo\PycharmProjects\Wolverine\static\pic\\"+date+'.jpg')

    p="/static/pic/"+date+'.jpg'

    pwd=random.randint(0000,9999)

    obj=Db()

    res=obj.insert("insert into login values('"+email+"','"+str(pwd)+"','police')")

    obj.insert("insert          into          traffic_police          VALUES
('"+str(res)+"','"+name+"','"+email+"','"+phone+"','"+police_station+"','"+str(p)+"','"+latitude
+"','"+longitude+"')")

    return '<script>alert("added
succesfully");window.location="/admin_home#contact"</script>'

```

```

@app.route('/add_signal')

def add_signal():

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    return render_template('admin/add_signal.html')

@app.route('/add_signal_post',methods=['post'])

def add_signal_post():

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    dis=request.form['select']

    place=request.form['textfield']

    latitude=request.form['textfield2']

    longitude=request.form['textfield3']

    obj=Db()

    obj.insert("insert                                     into
traffic_signal(place,district,latitude,longitude)values('"+place+"','"+str(dis)+"','"+latitude+"','"
+longitude+"')")

    return '<script>alert("addeed
succesfully");window.location="/admin_home#contact"</script>'

@app.route('/edit_police/<pid>',methods=['get'])

def edit_police(pid):

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    obj = Db()

    res=obj.selectOne("select * from traffic_police WHERE police_id='"+pid+"'")

```

```

return render_template('admin/edit_police.html',data=res, pid=pid)

@app.route('/edit_police_post',methods=['post'])

def edit_police_post():

    if session['lg']!='lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    pid=request.form['pid']

    Name=request.form['textfield']

    police_station=request.form['textfield2']

    image=request.files['fileField']

    phone=request.form['textfield3']

    date = datetime.datetime.now().strftime("%y%m%d-%H%M%S")

    image.save(r"C:\Users\anumo\PycharmProjects\Wolverine\static\pic\\" + date + '.jpg')

    p = "/static/pic/" + date + '.jpg'

    obj=Db()

    res=obj.selectOne("select * from traffic_police where name='"+Name+"' and
police_station='"+police_station+"' and image='"+str(p)+"' and phone='"+phone+"' and
police_id='"+pid+"'")

    print(res)

    if res is not None:

        return'<script>alert("no change");window.location="/admin_home#contact"</script>'

    else:

        if request.files!="none":

            if image.filename!="":

                obj = Db()

```

```

        obj.update("update traffic_police set name='"+Name+"',
police_station='"+police_station+"', image='"+str(p)+"', phone='"+phone+" where
police_id='"+pid+"'")

```

```

        print("update traffic_police set name='"+Name+"',
police_station='"+police_station+"', image='"+str(p)+"', phone='"+phone+" where
police_id='"+pid+"'")

```

```

        return'<script>alert(" updated");window.location="/view_police#contact"</script>'

```

```

    else:

```

```

        obj.update("update traffic_police set name='"+ Name + "', police_station='"+
police_station + "', phone='"+ phone + "' where police_id='"+ pid + "'")

```

```

        return'<script>alert(" updated");window.location="/view_police#contact"</script>'

```

```

    else:

```

```

        obj.update("update traffic_police set name='"+ Name + "', police_station='"+
police_station + "', phone='"+ phone + "' where police_id='"+ pid + "'")

```

```

        return ' <script>alert(" updated
succesfully");window.location="/view_police#contact"</script>'

```

```

@app.route('/edit_Place/<pid>',methods=['get'])

```

```

def edit_place(pid):

```

```

    if session['lg']!='lin':

```

```

        return '<script>alert("you are logged out");window.location="/"<script>'

```

```

    obj=Db()

```

```

    res = obj.selectOne("select * from important_place WHERE place_id='"+ pid + "'")

```

```

    return render_template("admin/edit_important_places.html",data=res,pid=pid)

```

```

@app.route('/edit_place_post',methods=['post'])

```

```

def edit_place_post():

```

```

    if session['lg']!='lin':

```

```

        return '<script>alert("you are logged out");window.location="/"<script>'

pid = request.form['pid']

District = request.form['select']

place_name = request.form['textfield']

place_type = request.form['select1']

description = request.form['textfield2']

image=request.files['fileField']

date = datetime.datetime.now().strftime("%y%m%d-%H%M%S")

image.save(r"C:\Users\anumo\PycharmProjects\Wolverine\static\pic\\" + date + '.jpg')

p = "/static/pic/" + date + '.jpg'

obj = Db()

res = obj.selectOne( "select * from important_place where place_name=" + place_name +
"" and district=" + District+ "" and place_type=" + place_type+ ""and description=" +
description+ "" and picture=" + str(p) + "" and place_id=" + pid + """)

if res is not None:

    return '<script>alert("no change");window.location="/admin_home#contact"</script>'

else:

    if request.files != "none":

        if image.filename != "":

            obj = Db()

            obj.update("update important_place set
place_name="+place_name+",district="+District+",place_type="+place_type+",descriptio
n="+description+" image="+str(p)+" where place_id="+str(pid)+"")

            return '<script>alert("updated
succesfully");window.location="/view_important_place#contact"</script>'

        else:

```

```

        obj.update("update important_place set place_name=" + place_name + ",district=" + District + ",place_type=" + place_type + ",description=" + description + " where place_id=" + str(pid) + "")

        return '<script>alert("updated successfully");window.location="/view_important_place#contact"</script>'

    else:

        obj.update("update important_place set place_name=" + place_name + ",district=" + District + ",place_type=" + place_type + ",description=" + description + " where place_id=" + str(pid) + "")

        return '<script>alert("updated successfully");window.location="/view_important_place#contact"</script>'

@app.route('/edit_signal/<pid>', methods=['get', 'post'])

def edit_signal(pid):

    if session['lg'] != 'lin':

        return '<script>alert("you are logged out");window.location="/"</script>'

    if request.method == "POST":

        district = request.form['select']

        place = request.form['textfield']

        latitude = request.form['textfield2']

        longitude = request.form['textfield3']

        obj = Db()

        obj.update("update traffic_signal set place=" + place + ",district=" + str(district) + ",latitude=" + latitude + ",longitude=" + longitude + " where signal_id=" + str(pid) + "")

        print("update traffic_signal set place=" + place + ",district=" + str(district) + ",latitude=" + latitude + ",longitude=" + longitude + " where signal_id=" + str(pid) + "")

```

```

        return '<script>alert("updated
succesfully");window.location="/view_signal#contact"</script>'

    else:

        obj=Db()

        res = obj.selectOne("select * from traffic_signal WHERE signal_id=" +str(pid) + "")

        return render_template("admin/edit_signal.html",data=res,pid=pid)

@app.route('/delete_police/<pid>')

def delete_police(pid):

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    # pid=request.form['pid']

    obj=Db()

    obj.delete("delete from traffic_police where police_id="+str(pid)+ "")

    return redirect('/view_police#contact')

@app.route('/delete_signal/<pid>')

def delete_signal(pid):

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    # pid=request.form['pid']

    obj=Db()

    obj.delete("delete from traffic_signal where signal_id="+str(pid)+ "")

    return redirect('/view_signal#contact')

@app.route('/delete_place/<pid>')

def delete_place(pid):

    if session['lg']!= 'lin':

```

```

        return '<script>alert("you are logged out");window.location="/"<script>'

# pid=request.form['pid']

obj=Db()

obj.delete("delete from important_place where place_id='"+str(pid)+"'")

return redirect('/view_important_place#contact')

@app.route('/view_device_allocation')

def view_device_allocation():

    if session['lg']!='lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    return render_template('admin/view_device_allocation.html')

@app.route('/view_important_place')

def view_important_place():

    if session['lg']!='lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    obj=Db()

    res=obj.select("select * from important_place")

    return render_template('admin/view_important_place.html',data=res)

@app.route('/view_police')

def view_police_police():

    if session['lg']!='lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    obj=Db()

    res=obj.select("select * from traffic_police")

    return render_template('admin/view_police.html',data=res)

```



```

@app.route('/view_signal')

def view_signal():

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    obj=Db()

    res=obj.select("select * from traffic_signal")

    return render_template('admin/view_signal.html ',data=res)

@app.route('/view_spot_complaint')

def view_spot_complaint():

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    obj=Db()

    res=obj.select ("select * from complaint")

    return render_template('admin/view_spot_complaint.html',data=res)

```

## 5.2 Traffic police

```

@app.route('/traffic_home')

def traffic_home():

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    return render_template("traffic_police/traffic_home.html")

@app.route('/view_emergency_alert')

def view_emergency_alert():

```

```

if session['lg']!= 'lin':

    return '<script>alert("you are logged out");window.location="/"<script>'

obj=Db()

res=obj.select ("select * from emergency_alert ")

return render_template('traffic_police/view_emergency_alert.html',data=res)

@app.route('/tra_view_spot_complaint')

def tra_view_spot_complaint():

    if session['lg']!= 'lin':

        return '<script>alert("you are logged out");window.location="/"<script>'

    obj=Db()

    res=obj.select ("select * from complaint,user where user.user_id=complaint.user_id")

    return render_template('traffic_police/view_spot_complaint.html',data=res)

@app.route('/logout')

def logout():

    session['lg']=" "

    return redirect('/')

@app.route("/aa")

def aa():

    return render_template("admin/index.html")

```

### 5.3 Main Section

```

@app.route('/help',methods=['post'])

def help():

    la=request.form['lati']

    lo=request.form['logi']

```

```

condi = request.form['cond']

print(la)

obj = Db()

obj.insert("insert into road_condition
values(",curdate(),curtime(),"+la+", ""+lo+", ""+condi+"")")

return jsonify(status="ok")

@app.route('/speedlimit',methods=['post'])
def speedlimit():

    sp = request.form['spdlmt']

    id = request.form['id']

    obj = Db()

    obj.insert("insert into overspeed values(", "" + id + ",curdate(),curtime()," + sp + ")")

    return jsonify(status="ok")

@app.route('/view_overspeed_limit',methods=['post'])
def view_overspeed_limit():

    obj = Db()

    qry=obj.select("select* from overspeed,user where overspeed.user_id=user.user_id ")

    return jsonify(status="ok" ,data = qry)

if __name__ == '__main__':

    app.run(debug=True,port=4000)

    # app.run(debug=True,port=4000,host="0.0.0.0")

```



## **6. SYSTEM TESTING**

### **6.1 TESTING AND EVALUATION**

Testing is a process of executing a program with the intent of finding an error. Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design and coding. Testing includes verifications of the basic logic of each program and verification that the entire system works properly. Testing demonstrates that software functions appear to be working according to specification. In addition, data collected as testing is conducted provides a good indication of software quality as a whole. The debugging process is the most unpredictable part of testing process. Testing begins at the module level and works towards the integration of the entire computer-based system testing and debugging are different activities, but any testing includes debugging strategy for software testing must accommodate low level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system function, against customer requirements. No testing is complete without verification and validation part. The goals of verification and validation activities are to

access and improve the quality of work products generated during the development and modification of the software. There are two types of verification: life cycle verification and formal verification. Life cycle verification is the process of determining the degree to which the products of the given phase of the development cycle fulfil the specification established during the prior process. Formal verification is the rigorous mathematical demonstration that source code confirms to its specifications. Validation is a process of evaluating software at the end of the software development process to determine compilation with the requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The primary objectives, when we test software are the following:

- Testing is a process of exceeding with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.
- A successful test is one uncovers undiscovered errors.

Thus, testing plays a very critical role in determining the reliability and efficiency of the software and hence is very important stage in software development. Tests are to be conducted on the software to evaluate its performance under a number of conditions. Ideally, it should so at the level of each module and also when all of them are integrated to from the completed system. Software testing is done at different levels.

## **6.2 TESTING STRATEGIES**

A strategy for software testing integrates software test case design method in to a well-planned series of steps that result in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote a customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project progresses.

**The general characteristics of software testing strategies are:**

- Testing begins at the component level and works “outward” toward the integration of the entire computer system.
- Different testing techniques are appropriate at different points in time.

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

A strategy must provide guidance for the practitioner and set of milestones for the manager. Because the step on the test strategy occurs at a time when deadline pressure begins to rise, progress must be measurable and problem must surface as early as possible.

The software team's approach to testing is defining a plan that describes an overall strategy and a procedure that defines specific testing steps and tests that will be conducted. In the proposed system, if the administrator makes any attempt to login to the application without entering his password, then the system will not allow the user to login to the application.

## **6.3 TESTING TECHNIQUES**

The various testing techniques are given below:

### **6.3.1 WHITE BOX TESTING**

White-box testing is also called as glass-box testing, is a test case design method that goes to the control structure of the procedural design to derive

test cases. Using white box testing methods, the software engineer can derive test cases that,

- ✓ Guarantee that all independent paths within a module have been exercised at Least once.
- ✓ Exercise all logical decision on their true and false sides.
- ✓ Execute all loops at their boundaries and within their operational sides.
- ✓ Exercise internal data structure to ensure their validity.

White box testing was successfully conducted on our system. All independent paths within a module have been executed at least once and all logical decisions have been exercised on their true and false sides.

### **6.3.2 BLACK BOX TESTING**

Black-box testing is also called as behavioural testing, focuses on the functional requirement of the software. It is a complimentary approach that is likely uncover a different class of

errors than white box methods. Black box testing attempts to find errors in the following categories.

- ✓ Incorrect or missing functions.
- ✓ Interface errors.
- ✓ Error on data structures or external database access.
- ✓ Behaviour or performance errors.
- ✓ Initialization and termination errors.

Black box testing was successfully conducted on your system. The system was divided into a number of modules and testing was conducted on each module. We have tested the system for incorrect or missing functions, interface and performance errors.

### **6.3.3 UNIT TESTING**

Unit testing comprises the set of tests performed by an individual programmer prior to the integration of the system. Testing removes residual bugs and improves the reliability of the system.

Testing allows the developer to find out the design faults if any, and enable correction if needed. Exhaustive unit testing has to be carried out to ensure the validity of the data. In order to successfully test the entire package, unit testing is carried out. Each module was tested as when it was developed. Thus it proved easier to conduct minute testing operation and correct them then and there.

### **6.3.4 INTEGRATION TESTING**

Bottom-up integration is the traditional strategy used to integrate the component of a software system into a functional whole. Bottom-up integration consists of unit testing, followed by subsystem testing and followed by testing of the entire system. Unit testing has the goal of discovering errors in the individual parts of the system.

Parts are tested in isolation from one another in an artificial environment known as “Test Harness”, which consists of driver programs and data necessary to exercise the modules. Unit testing should be as exhaustive as possible to ensure that each representative case handled by each module has been tested. Unit testing is eased by a system structure that is composed of small loosely coupled modules.



Both control and data interfaces must be tested. Large software system may require several levels of subsystem testing. Lower level subsystems are successively combined to form higher level subsystems. In most software systems, exhaustive testing of subsystem capabilities is not feasible due to the combination complexity of the module interfaces. Therefore, test cases must be carefully chosen to exercise the interfaces in the desired manner.

### **6.3.5 ACCEPTANCE TESTING**

Acceptance testing involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements. It is not unusual for two sets of acceptance test to be run, those developed by the quality group and those developed by the customer.

In addition to the functional and performance tests, stress tests are performed to determine the limitation of the system. For example, a compiler might be tested to determine the effect of the symbol table overflow, or real-time system might be tested to determine the effect of simultaneous arrival of numerous high priority interrupts.

### **6.3.6 OUTPUT TESTING**

Output testing of the proposed system is important since no system could be useful if it does not produce the required output.

The output format on the screen is found to be correct, as the format was designed in the system design phase according to the user needs. For the hard copy also the output comes out as the specified requirements by the user. Hence output testing doesn't result in any correction on the system.

## **7.SYSTEM IMPLEMENTATION AND DEPLOYMENT**

Implementation is the process of deploying the new system in the operational environment. Proper implementation is essential to provide a reliable system to meet the organizational requirement. There are four types of implementation methods. They are Direct Changeover, Phased Implementation, Parallel Run and Pilot Approach. The most commonly used implementation methods are Pilot Approach and Parallel Run.

The system which is developed as a web application hence the other functions as normal application, as usual some web development technologies are used in the implementation of the project. The language I selected to program this software is PHP. The reason I selected PHP is that is a simple and powerful language that especially developed to create web application.

Technologies used in the development of the software are:

- ✓ Programming language – Python
- ✓ DBMS – MySQL server
- ✓ Development tool – Py charm
- ✓ Development platform – Windows 10

The front end is HTML, and CSS and back end is MySQL Server and Python. The system developed on Pycharm in Windows 10 operating system.

## **8. CONCLUSION**

The WOLVERINE application is implemented for to identify surface disruptions based on accelerometer patterns and also use a crowd sourcing technique for store or pass incident details to the server. Sensor networks together with surveillance cameras help identify elements disturbing cars' mobility. On the contrary, in developing countries, the lack of this level of infrastructure makes the task of roads' maintenance and traffic control challenging. The presence of roadway surface disruptions (RSDs), in particular, affects the economy and reputation of individuals and companies. Urban computing strategies can be adopted to collect data about the presence of RSDs and be applied to monitor traffic related issues.

## **8.1 FUTURE ENHANCEMENT**

As a future venture, it is suggested to make some changes to provide more services. In future we will be including the 360 degree street view, real-time traffic, directions, location searching, and route planning for walking, biking, driving, or public transportation.

All the functions have been done carefully and successfully in the software, and if any development is necessary in future, it can be done without affecting the design by adding additional modules to the system.

## **9. REFERENCE**

### **Online Reference**

- Google
- Youtube

### **Books Reffered**

- Python programming:A beginner's guide to learn python from zero
- Learn android development with android studio

### **Websites**

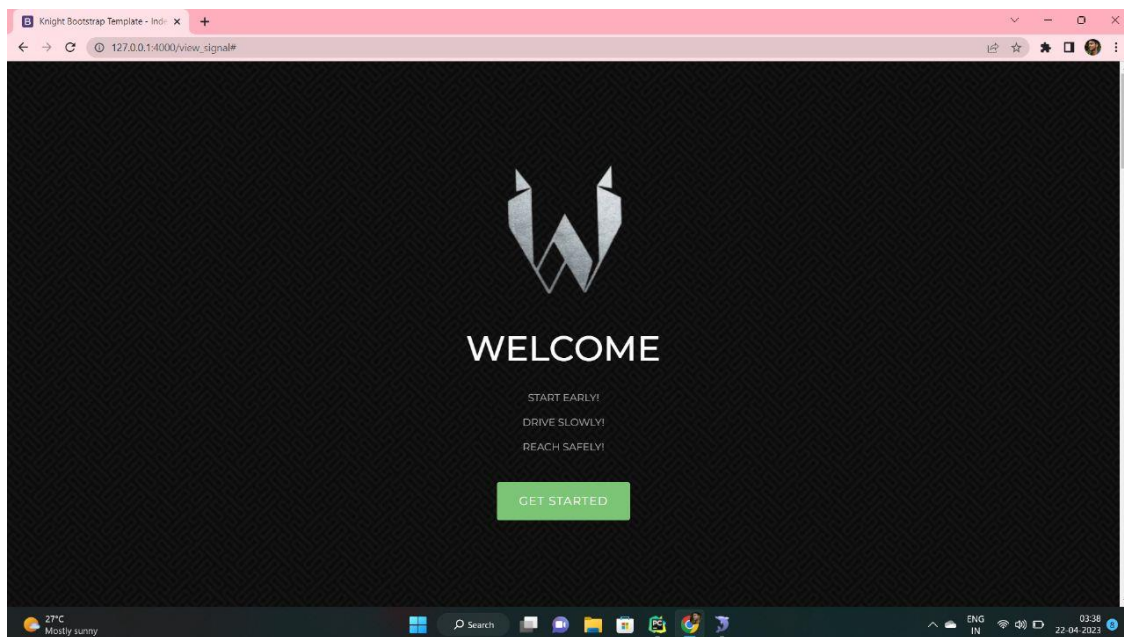
- W3Schools.com
- Python.org
- Stackoverflow.co

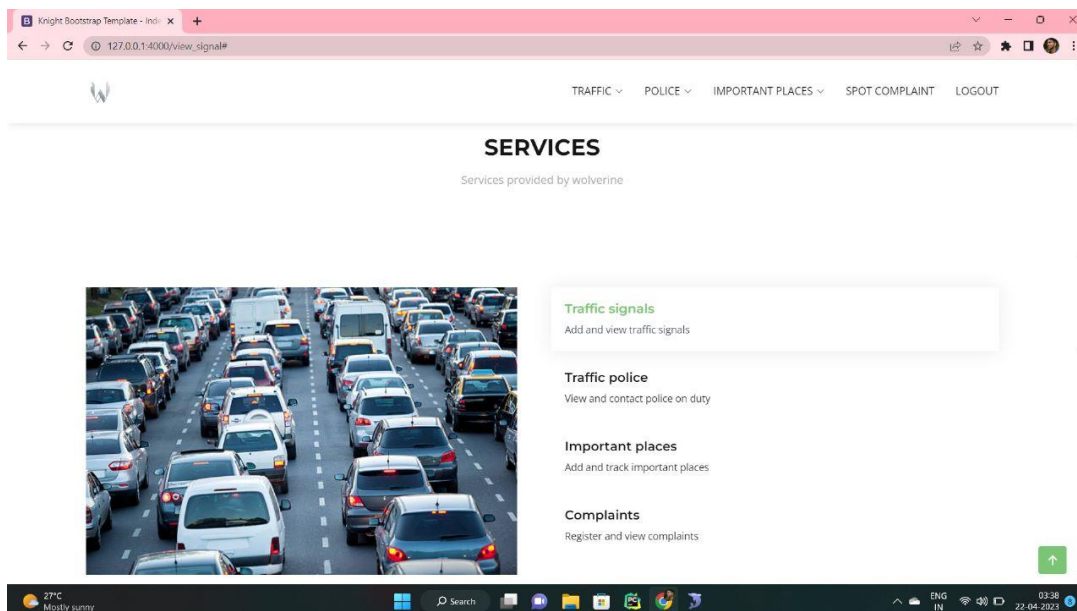
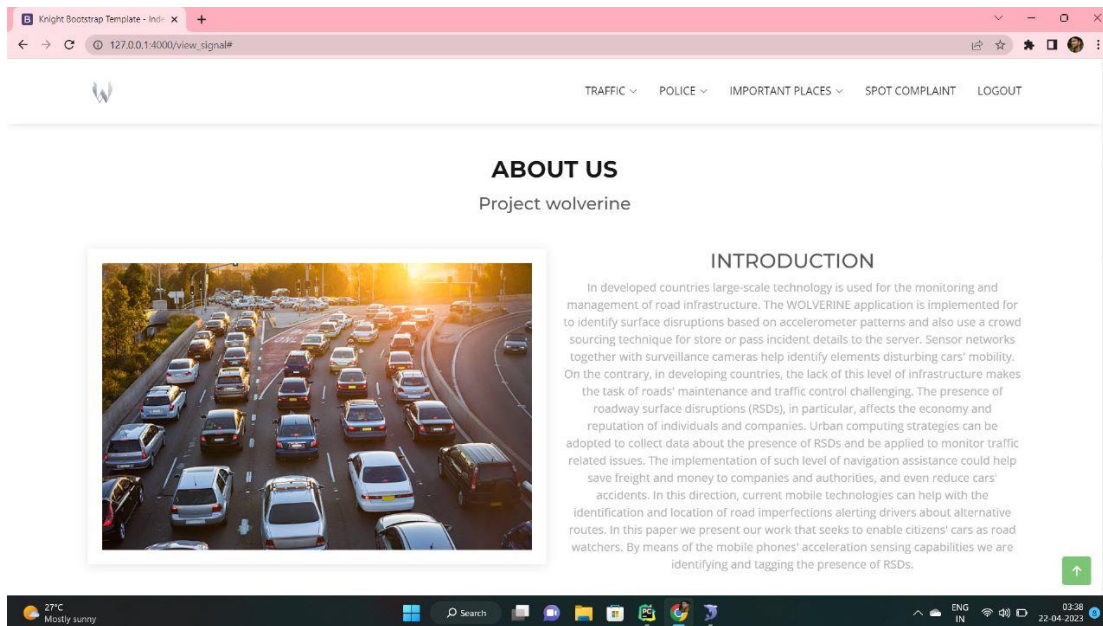
## **10. APPENDIX**

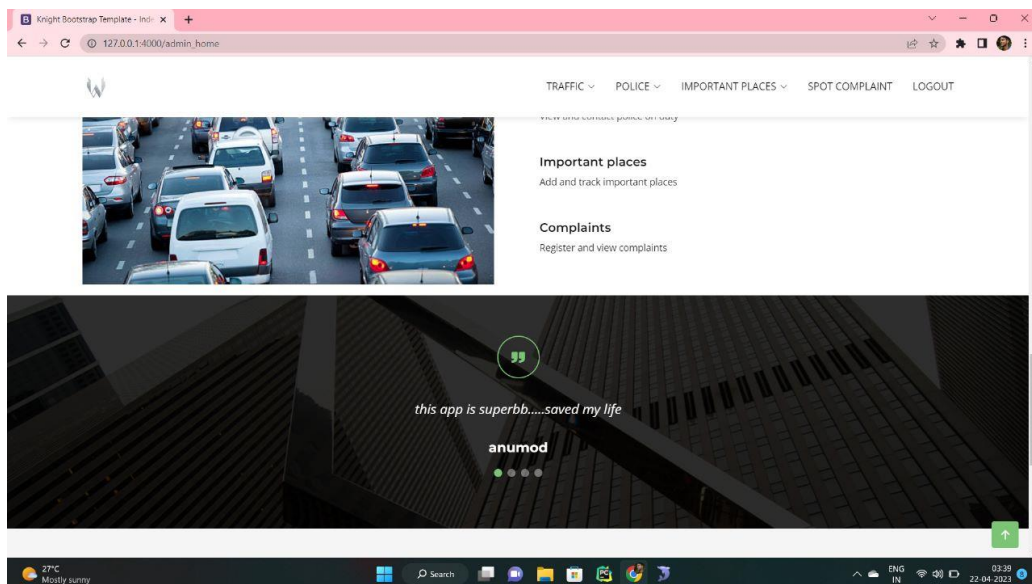
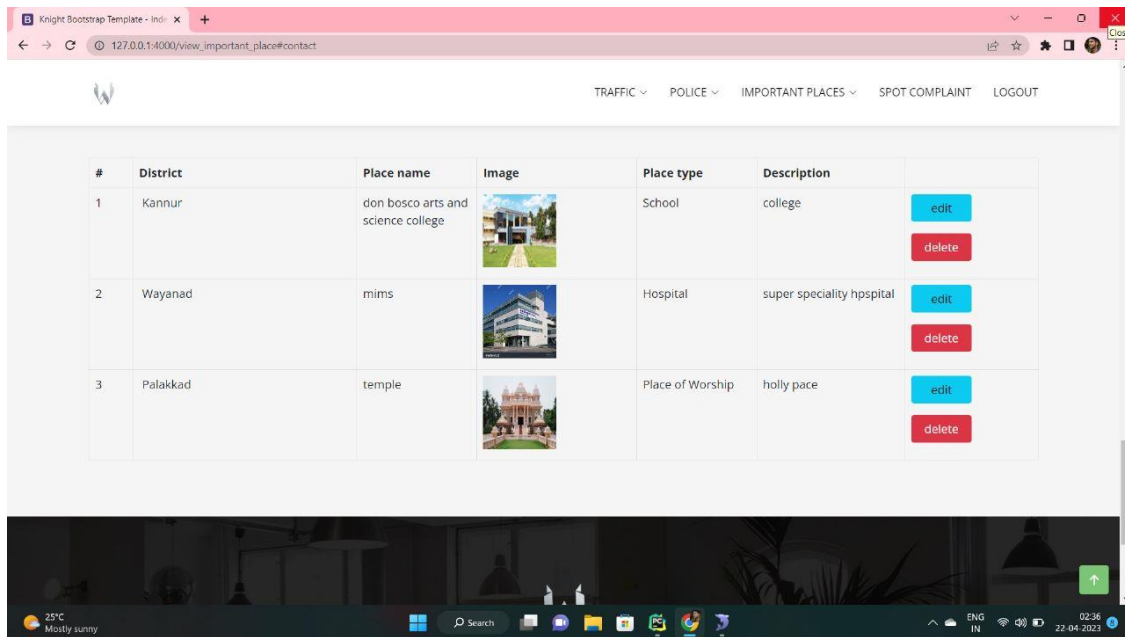


## 10.1 Screenshots

### *10.1.1 Homepage:*







Knight Bootstrap Template - Indi x police photo - Google Search x Mumbai Police Commissioner Hi x police.webp (1230x757) x +

127.0.0.1:4000/view\_signal#contact

TRAFFIC POLICE IMPORTANT PLACES SPOT COMPLAINT LOGOUT



#	District	place	
1	Kannur	kakkadd	<div>delete</div> <div>track</div> <div>edit</div>
2	Kannur	nadal	<div>delete</div> <div>track</div> <div>edit</div>
3	Kasaragod	Athirapilly	<div>delete</div> <div>track</div> <div>edit</div>

27°C Mostly sunny 03:52 22-04-2023

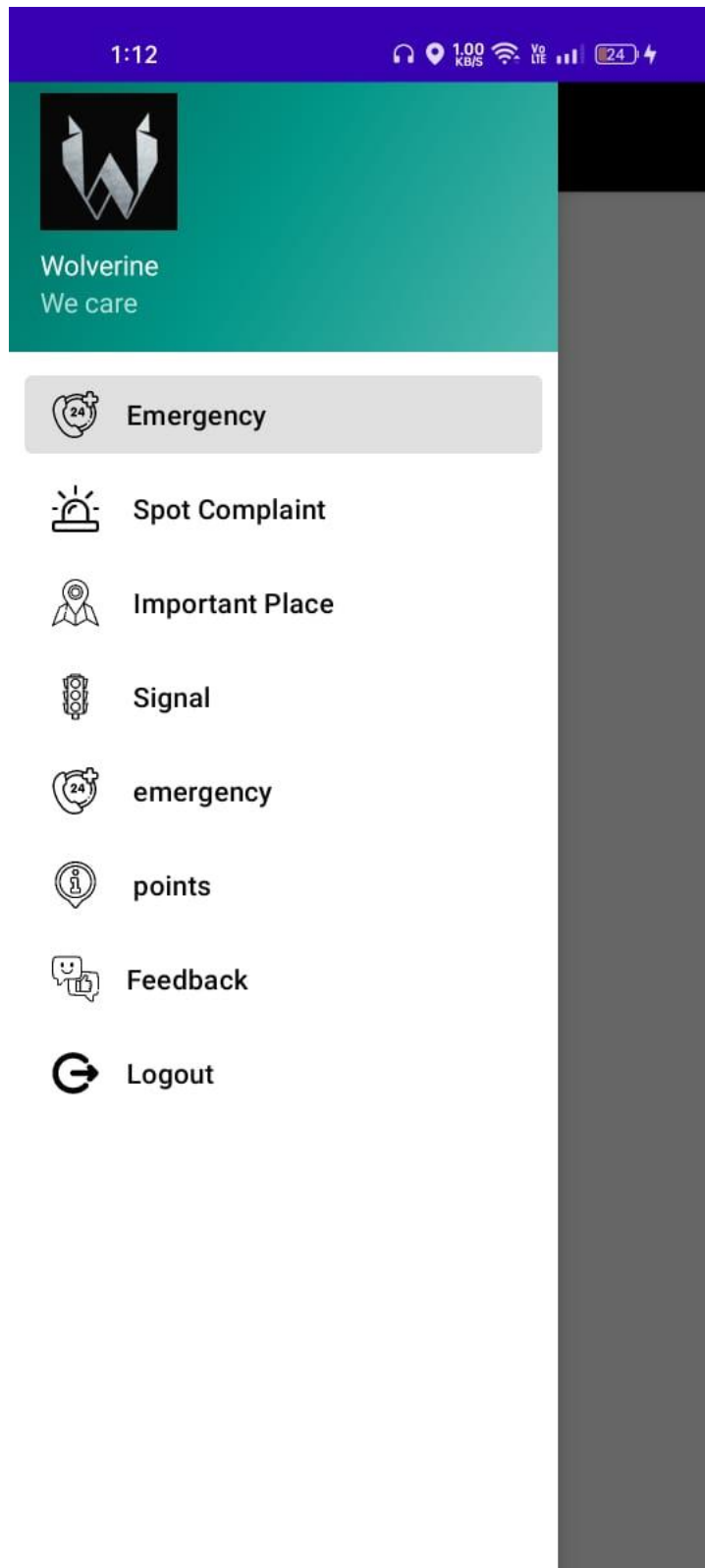
Knight Bootstrap Template - Indi x police photo - Google Search x Mumbai Police Commissioner Hi x police.webp (1230x757) x +

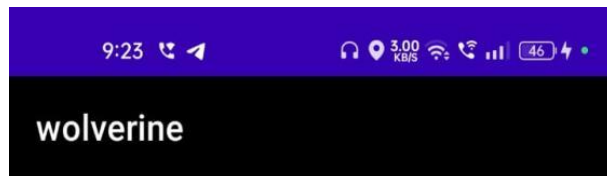
127.0.0.1:4000/view\_police#contact

TRAFFIC POLICE IMPORTANT PLACES SPOT COMPLAINT LOGOUT

#	Name	Police station	Image	Phone	email	
1	shivadasan	mowwanchery		8890800808	a@gmail.com	<div>edit</div> <div>delete</div> <div>track</div>
2	ram	chakkarakal		2147483647	ram@gmail.com	<div>edit</div> <div>delete</div> <div>track</div>

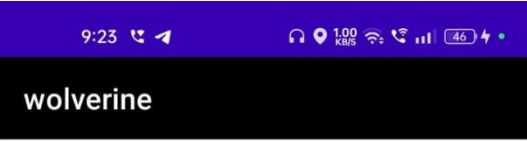
27°C Mostly sunny 03:52 22-04-2023





Text

SEND



Place **kakkadd**

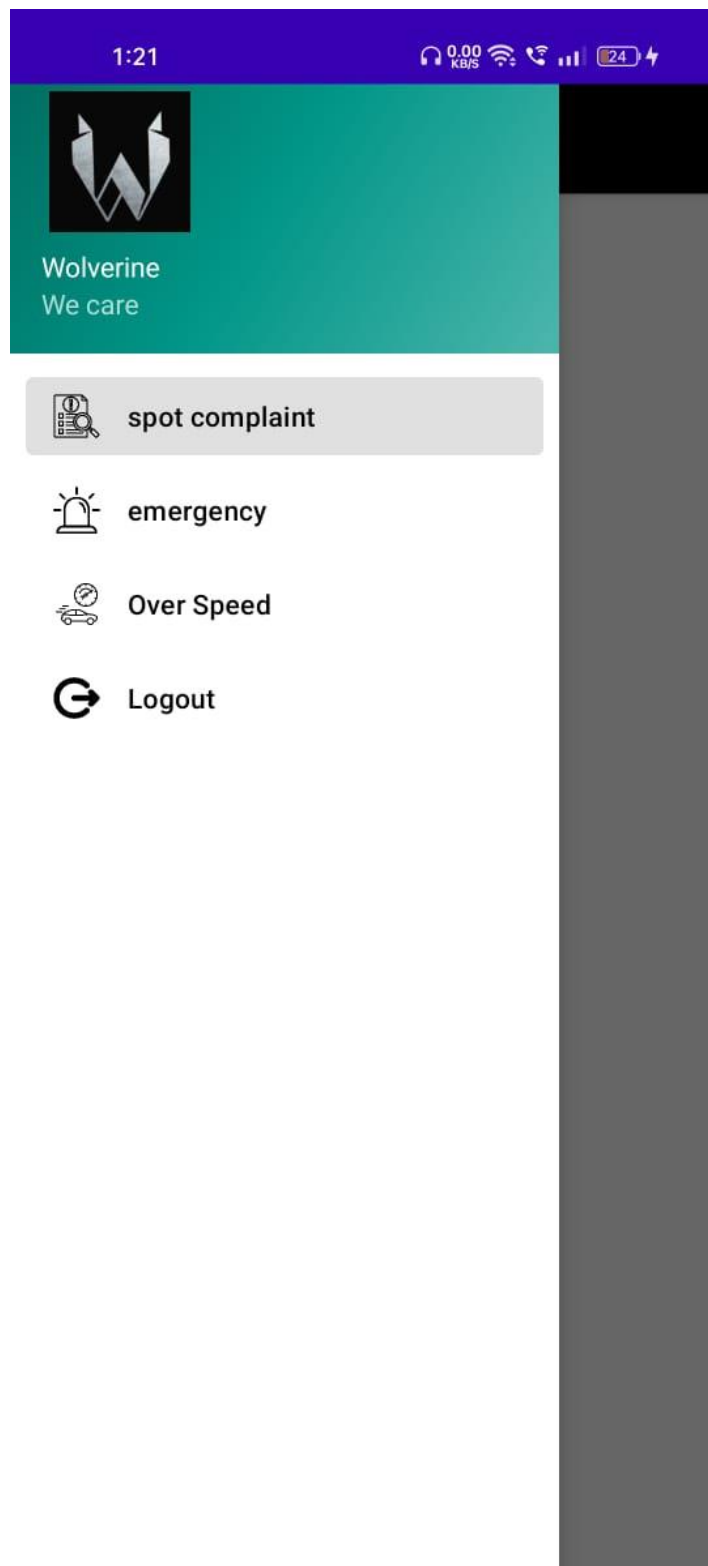
District Kannur

TRACK

Place **thana**

District Wayanad

TRACK





1:21

0.72 KB/s

## wolverine



Date 12/12/22

Time 12:56

Content unknown car parked in front of my gate

User 2

TRACK



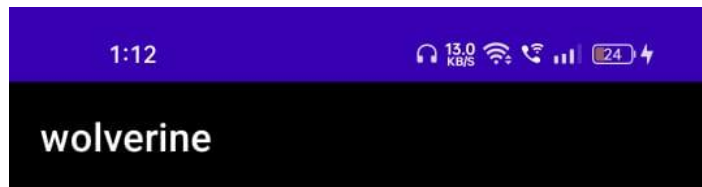
Date 12/2/22

Time 12:52

Content road disruption due to bike accident

User 1

TRACK



Place Name don bosco arts and sci

District Kannur

Place Type School

Discription college

SEND IMPORTANT POINTS



Place Name mims

District Wayanad

Place Type Hospital

Discription super speciality h

SEND IMPORTANT POINTS



Place Name temple

District Palakkad

Place Type Place of Wors

Discription holly pace



# **A PROJECT REPORT ON AI SKIN EXPERT**

Submitted in partial fulfillment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

**By**

**ARJUN P P**

**Reg.No:DB20BCAR07**

**SHANITH SHAJI**

**Reg.No:DB20BCAR12**

**AKSHAY BIJU**

**Reg.No:DB20BCAR19**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**

# **A PROJECT REPORT ON AI SKIN EXPERT**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ARJUN P P**

**Reg.No:DB20BCAR07**

**SHANITH SHAJI**

**Reg.No:DB20BCAR12**

**AKSHAY BIJU**

**Reg.No:DB20BCAR19**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**

**DON BOSCO ARTS & SCIENCE COLLEGE**  
**ANGADIKADAVU**  
**IRITTY, KANNUR**



**CERTIFICATE**

*Certified that this report titled AI SKIN EXPERT is a bonafide record of the project work done by **Arjun P P(Reg.No:DB20BCAR07)**, **Shanith Shaji(DB20BCAR12)** and **Akshay Biju(DB20BCAR19)** under the supervision and guidance ,towards partial fulfilment of the requirement for award of the degree of bachelor of computer application (BCA) of the Kannur university.*

Project Guide:

Head of the Department:

Angadikadavu

External Examiner

Date:

1.

2.



## **Declaration**

We ARJUN.P.P, SHANITH SHAJI and AKSHAY BIJU sixth semester BCA student of Don Bosco Arts & Science College, Angadikadavu, under Kannur University do hereby declare that the project entitled “**AI SKIN EXPERT**” is the original work carried out by me in the sixth semester under the supervision of Ms.Sindhu P.M, Lecturer of the Department of BCA, Don Bosco Arts & Science College, Angadikadavu, in partial fulfillment of the requirement for the award of the degree Bachelor of Computer Application, Kannur University.

Angadikadavu :

Date :

ARJUN P P

SHANITH SHAJI

AKSHAY BIJU



## **ACKNOWLEDGEMENT**

First of all we thank the lord almighty for his immense grace and blessings showered on me at every stages of this work. I am greatly indebted to our Principal Fr. Dr. Francis Karackat SDB, Don Bosco Arts & Science College, Angadikadavu for providing the opportunity to take up this project as part of my curriculum.

We deeply indebted to my project guide Ms. Sindu PM, lectures of department of BCA, for her assistance and valuable suggestions as guide. She made this project a reality.

We express our sincere thanks to Mrs. Sruthi Nimesh , Mr. Hebin Layola, Mrs. Fincy Cyriac and Mrs. Vineetha Mathew, lecturers of department of BCA, for their valuable suggestions during the course of this project. Their critical suggestions helped me to improve the project work.

Acknowledging the efforts of everyone, their chivalrous help in the course of the project preparation and their willingness to collaborate with the work, their magnanimity through lucid technical details lead to the successful completion of my project.

We would like to express my sincere thanks to all my friends, colleagues, parents and all those who have directly or indirectly assisted during this work.

# CONTENTS

chapters	contents	Page No
1	Introduction	1
1.1	Model	3
2	System Analysis	7
2.1	Existing System	8
2.1.1	Disadvantage Of Existing System	8
2.2	Proposed System	8
2.2.1	Advantage Of Proposed System	9
2.3	Feasibility Study	10
2.3.1	Economic Feasibility	10
2.3.2	Technical Feasibility	11
2.3.3	Behavioural Feasibility	11
2.4	System Specification	11
2.4.1	Software Specification	12
2.4.2	Hardware Specification	12
2.5	Identification Of Actors	13
2.6	Identification Of Use Cases	13
2.6.1	Use Cases For The Actor Administrator	14
2.6.2	Use Cases For The Actor Doctor	14
2.6.3	Use Cases For The Actor Patients	15
2.6.4	Use Case Diagram	16

3	SYSTEM DESIGN	18
3.1	Introduction	19
3.2	Database Design	19
3.3	Table Design	20
3.4	AI Skin expert	21
3.5	Data Flow Diagram	24
3.6	ER Diagram	33
4	CODING	35
4.1	Input Interface	36
4.2	Output Interface	36
4.3	Software Description	36
4.4	HTML	36
4.4.1	CSS	38
4.4.2	JavaScript	40
4.4.3	Android	41
4.4.4	JAVA	45
4.4.5	Python	47
4.4.6	MySQL	47
4.4.6	Flask	49
5	CODING PAGES	50
5.1	Admin Page	51
5.2	Login page	52
5.3	Patient signup	55
5.4	View_doctor	58
5.5	Login.html	61

5.6	Index.html	62
5.7	Approve_doctor.html	73
5.8	Index.html	74
6	System Testing	89
6.1	Testing And Evaluation	90
6.2	Testing Strategies	91
6.3	Testing Techniques	92
6.3.1	White Box Testing	92
6.3.2	Black Box Testing	92
6.3.3	Unit Testing	93
6.3.4	Integration Testing	93
6.3.5	Acceptance Testing	93
6.3.6	Output Testing	94
7	SYSTEM IMPLEMENTATION AND DEPLOYMENT	95
8	CONCLUSION	96
9	REFERANCE	97
10	APPENDIX	98
10.1	Screenshots	99

# **1.INTRODUCTION**

## **1.1 Project Overview**

The AI Skin Expert is an innovative mobile application designed to help users identify skin conditions and diseases using the convolutional neural network which is the highly trusted methodology for prediction systems along with symptom based disease prediction systems.

. The app utilizes machine learning algorithms to analyse photographs of the user's skin and provide a diagnosis of potential skin issues.

When a user takes a photo of their skin using the app, the image is processed and analysed by the AI algorithm. The app then uses this information to provide a potential diagnosis of the skin condition or disease based on its extensive database of skin disorders. Users will have the option to either consult a doctor online or to self-medicate based on the diagnosis provided by the AI Skin Expert app.

The AI Skin Expert app can identify a wide range of skin conditions including skin cancer, eczema, allergies and more. Additionally, the app provides detailed information on each condition, including its symptoms, causes, and potential treatment options.

The application also includes a feature that allows users to chat with the doctors. This is particularly useful for users who are undergoing treatment for a skin condition, as they can monitor their progress and identify whether their treatment is working effectively.

Overall, the AI Skin Expert app is a powerful tool that empowers users to take control of their skin health. By leveraging the power of AI, the app provides users with accurate and reliable information about their skin conditions, helping them make informed decisions about their health and wellness.

## **NEED FOR THE SYSTEM**

Skin conditions and diseases are common worldwide. Early detection and diagnosis of skin conditions are essential for effective treatment and management. However, many people may not have access to dermatologists or may delay seeking medical attention due to the cost or inconvenience.

An AI Skin Expert app can help bridge this gap by providing users with an immediate diagnosis of potential skin conditions or diseases, without the need for a physical visit to a dermatologist. The app can analyze images of the skin using advanced AI algorithms and machine learning techniques, providing users with detailed information about each skin condition, including symptoms, causes, and potential treatment options.

The app can also store user data and analysis results in a secure and reliable database, allowing users to access previous analysis results and track any changes in their skin over time. By providing users with an affordable and accessible alternative to traditional methods of diagnosing skin conditions, the AI Skin Expert app can help improve early detection rates and ultimately lead to better health outcomes.

## **OBJECTIVES AND SCOPE**

In this new era, as people get busier, AI Skin Expert app is to provide an affordable and accessible alternative to traditional methods of diagnosing skin conditions. The app's primary goal is to help users detect and diagnose potential skin conditions or diseases at an early stage, allowing for prompt and effective treatment. The app will achieve this objective by analysing images of the skin using advanced AI algorithms and machine learning techniques, providing users with detailed information about each skin condition, including symptoms, causes, and potential treatment options.

The scope of the AI Skin Expert app is to provide users with an affordable and accessible tool to detect and diagnose potential skin conditions or diseases at an early stage by analysing images of the skin using advanced AI algorithms and machine learning techniques and providing users with detailed information about each skin condition.

The scope of the AI Skin Expert app includes the following features:

- 1) **Skin Image Analysis:** The app will allow users to capture images of their skin using a device camera and analyze those using advanced AI algorithms and machine learning techniques to detect potential skin conditions or diseases.
- 2) **Diagnosis:** The app will provide users with an immediate diagnosis of potential skin conditions or diseases based on the analysis of the skin images.

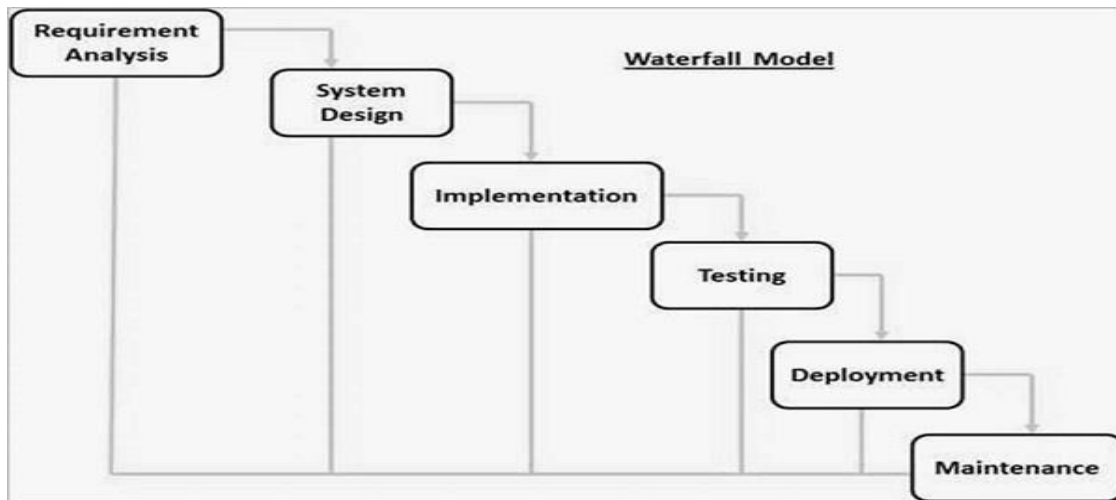
- 3) **Information:** The app will provide users with detailed information about each skin condition, including symptoms, causes, and potential treatment options.
- 4) **User Data Storage:** The app will store user data and analysis results in a secure and reliable database.
- 5) **User-Friendly Interface:** The app will have an intuitive and user-friendly interface that is easy to navigate and understand for individuals with varying levels of technical expertise.
- 6) **Compatibility:** The app will be compatible with various devices, including smartphones, tablets, and computers, and various operating systems, like Android.

## **MODEL:**

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially

### **Waterfall Model - Design:**

In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. Following is the pictorial representation of Iterative and Incremental model:



The sequential phases in Waterfall model are –

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.



All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

### **Waterfall Model - Application:**

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are –

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

### **The advantages of the waterfall model SDLC Model are as follows:**

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Some of the major advantages of the Waterfall Model are as follows –

- Simple and easy to understand and use

- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

**The disadvantages of the waterfall model SDLC Model are as follows:**

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows –

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang" at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

## **2. SYSTEM ANALYSIS**

## **Introduction**

System analysis is the process of collecting and interpreting facts, understanding problems and using the information to suggest improvement on the system. This will help to understand the existing system and determine how computers make its operation more effective. The aim of this analysis is to collect detailed information on the system and the feasibility study of the proposed system.

## **2.1 EXISTING SYSTEM**

Currently, diagnosing skin conditions and diseases requires a visit to a dermatologist or skin specialist. This process can be time-consuming, expensive, and inconvenient for many people. Additionally, there may be a shortage of dermatologists in certain areas, making it difficult for some individuals to receive timely care.

### **2.1.1 The Disadvantages of Existing System**

The existing system for diagnosing skin conditions and diseases typically involves visiting a dermatologist or skin specialist. While this approach can be effective, it has several disadvantages, including:

- **Time-consuming:** Visiting a dermatologist or skin specialist can be time-consuming, especially if the individual lives in a remote area or has a busy schedule. This can lead to delays in receiving a diagnosis and treatment.
- **Expensive:** The cost of visiting a dermatologist or skin specialist can be expensive, particularly for those without health insurance or with limited financial resources.
- **Inconvenient:** Some people may find it inconvenient to visit a dermatologist or skin specialist, especially if they live far away from a clinic or hospital.
- **Shortage of dermatologists:** In some areas, there may be a shortage of dermatologists or skin specialists, which can make it difficult for individuals to receive timely care.
- **Limited accessibility:** The existing system may not be accessible to everyone, particularly those who live in rural or remote areas.

## **2.2 PROPOSED SYSTEM**

The AI Skin Expert app aims to address these challenges by providing users with a convenient, accessible, and cost-effective tool for identifying skin conditions and diseases. With the app, users can take a photo of their skin and receive an immediate diagnosis of potential skin issues.

The app uses advanced AI algorithms to analyze the photo and provide a potential diagnosis of the skin condition or disease. Additionally, the app provides detailed information on each condition, including its symptoms, causes, and potential treatment options. This can help users make informed decisions about their skin health and wellness.

One of the key advantages of the AI Skin Expert app is its ability to reach a wider audience. By leveraging the power of AI, the app can provide skin diagnoses to individuals in areas where there may be a shortage of dermatologists or skin specialists.

Furthermore, the app can help to reduce the burden on dermatologists and skin specialists by providing an initial diagnosis of skin conditions. This can help to prioritize patients who need immediate attention and reduce wait times for those who require in-person consultations.

### **2.2.1 Advantages of Proposed System**

- Affordable and accessible alternative to traditional methods of diagnosing skin conditions.
- Helps detect and diagnose potential skin conditions or diseases at an early stage.
- Uses advanced AI algorithms and machine learning techniques to analyze skin images.
- Provides users with immediate diagnosis and detailed information about each skin condition.
- Stores user data and analysis results in a secure and reliable database.
- User-friendly interface for individuals with varying levels of technical expertise.
- Compatible with various devices and operating systems.

## **2.3 Feasibility Study**

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioural feasibilities.

The proposed system is theoretically investigated to check the feasibility and found that they are more reliable and efficient in the cases given below. There are three aspects in the feasibility study portion of the preliminary investigation.

✓ Economic feasibility

✓ Technical feasibility

✓ Behavioural feasibility

The proposed system must be evaluated from a technical point of view first, and if technical feasible their impact on the organization must be assessed. If compatible, the operational system can be devised. Then they must be tested for economic feasibility.

### **2.3.1 Economic Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors which affect the development of a new system is the cost it would require. Since the system developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

### **2.3.2 Technical Feasibility**

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, procedures and staff. Having identified an outline system, the investigation must go on suggest the type of equipment, required method developing the system, of running the system once it has been designed. The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Though the technology become obsolete after some period of time, due to the fact that newer version of some software supports older versions, the system still be used. So there are only minimal constraints involved with this project. The system has been developed using Python and .Android, along with the database software SQL server, thus we could conclude that the project is technically feasible for development.

### **2.3.3 Behavioural Feasibility**

People are inherently resistant to change and computers have been known to facilitate change. The System is designed in user friendly manner and we need to provide any special training for the persons using this software. The operating system used is Windows 10, which is also user friendly. Since the application is web biased and can easily accessed in a web browser, which is quite familiar to the intended users, it does not have any operational barriers. So no need to provide any special training for using this application software and hence it is behaviourally feasible.

## **2.4 System Specifications**

System Specification deals with the technical aspects the project has to meet in minimum to work successfully. This also includes the different aspects the software requirement is determined from. The technical details typically include:

- Software Specification
- Hardware Specification

### **2.4.1 Software Specifications**

The software required for the application depends on the following factors:

- ✓ The flexibility of the software
- ✓ Software contracts
- ✓ Limitation of the software

### **Software Requirement**

This specifies the minimum software requirements for implementing the system. This includes:

- Operating system : Window 7 or above
- Front End : JAVA( Android), Python
- Back end : My SQL
- Programming language : Java(Android), Python
- Browser : Chrome / Brave

### **2.4.2 Hardware Specifications**

The software required for the application depends on the following factors:

- ✓ Determining size and capacity requirements.
- ✓ Computer evaluation and measurement.
- ✓ Financial factors.
- ✓ Maintenance and support.

### **Hardware Requirement**

- Microprocessor : INTEL DUAL CORE or above
- Ram : 4 GB and above
- Hard disk : 160 GB and above
- Output Devices : Monitor
- Keyboard : standard keyboard



- Mouse : Standard mouse
- Connectivity : LAN & Wi-Fi

## 2.5 Identification of Actors

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yields an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

We can identify the actors through a list of questions. The answers to these questions bring out the actors of the system is.

- Admin
- Doctor
- Patients

Here we need to specify the use cases of each actor.

## 2.6 Identification of use cases

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yield an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

To find out the use cases, ask the following questions to each of the actors.

- ✓ Which functions does the actor require from the system? What does the actor need to do?
- ✓ Does the actor need to read, create, destroy, modify or store some kind of information in the system?
- ✓ Does the actor have to calculate something? And want to provide information for others?
- ✓ Could the actor's daily work be simplified or made more efficient by adding new functions to the system (typically functions which are currently not automated in the system)?

### **2.6.1 Use cases for the actor Administrator**

#### **1) Login:**

- i) The first step involved is login. The admin can login to the website using username and password.

#### **2) Doctor Management.**

- i) Approve Doctor
- ii) View Approved Doctors

#### **3) Disease Management**

- i) Add Diseases
- ii) View/Edit/Delete Diseases
- iii) Symptom Managements

#### **4) View Users**

#### **5) View Feedback**

#### **6) View doctor reviews**

### **2.6.2 Use cases for the actor Doctors**

#### **1) Login:**

- i) The Doctor can login to the website using username and password.

#### **2) Account Creation**

#### **3) Login**

#### **4) View Profile/Edit Profile/ Change Password**

#### **5) View Diseases**

#### **6) Predict Diseases using image**

#### **7) Predict Diseases using Symptoms**

#### **8) Schedule Management**

#### **9) View Patients appointments**

#### **10) Chatting with Patients**

#### **11) View Reviews**

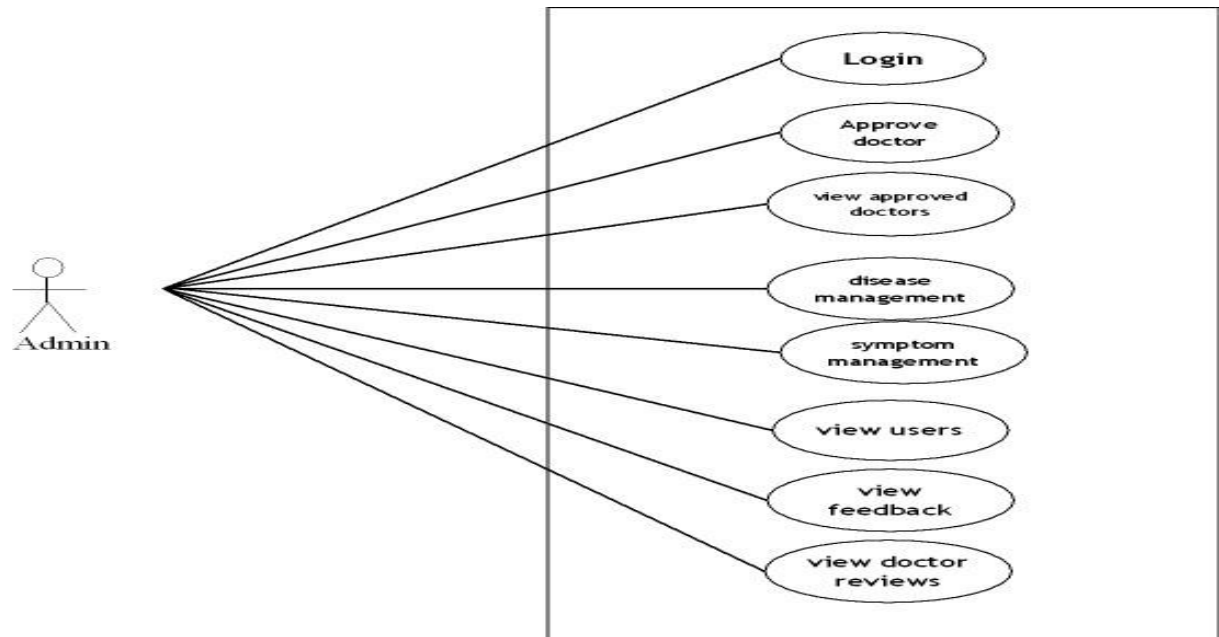
### **2.6.3 Use cases for the actor Patients**

**1) Login:**

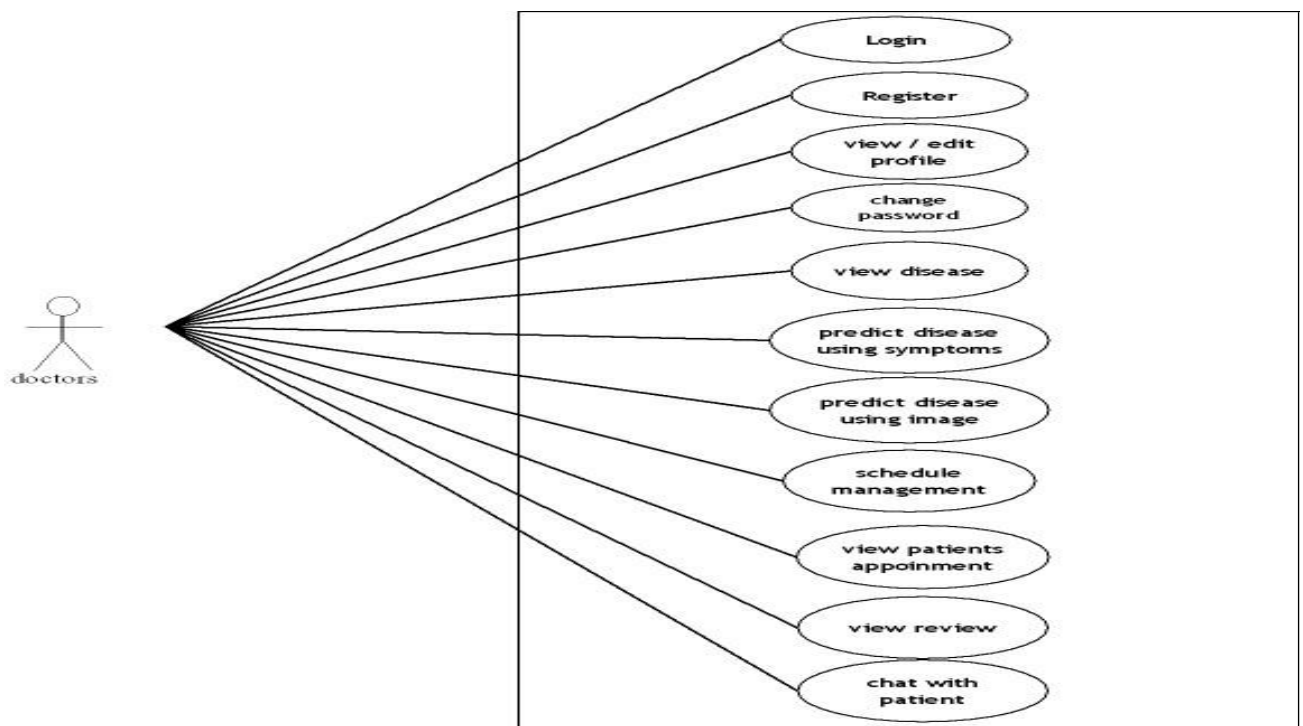
- i) Patients can login to the website using username and password.
- 2) Signup
- 3) Login
- 4) View Profile
- 5) View Doctors
- 6) View Doctors Schedule
- 7) Make doctor's appointment
- 8) Predict diseases by uploading photo
- 9) Predict diseases using Symptoms
- 10) Send feedback.

## 2.6.4 USE CASE DIAGRAM

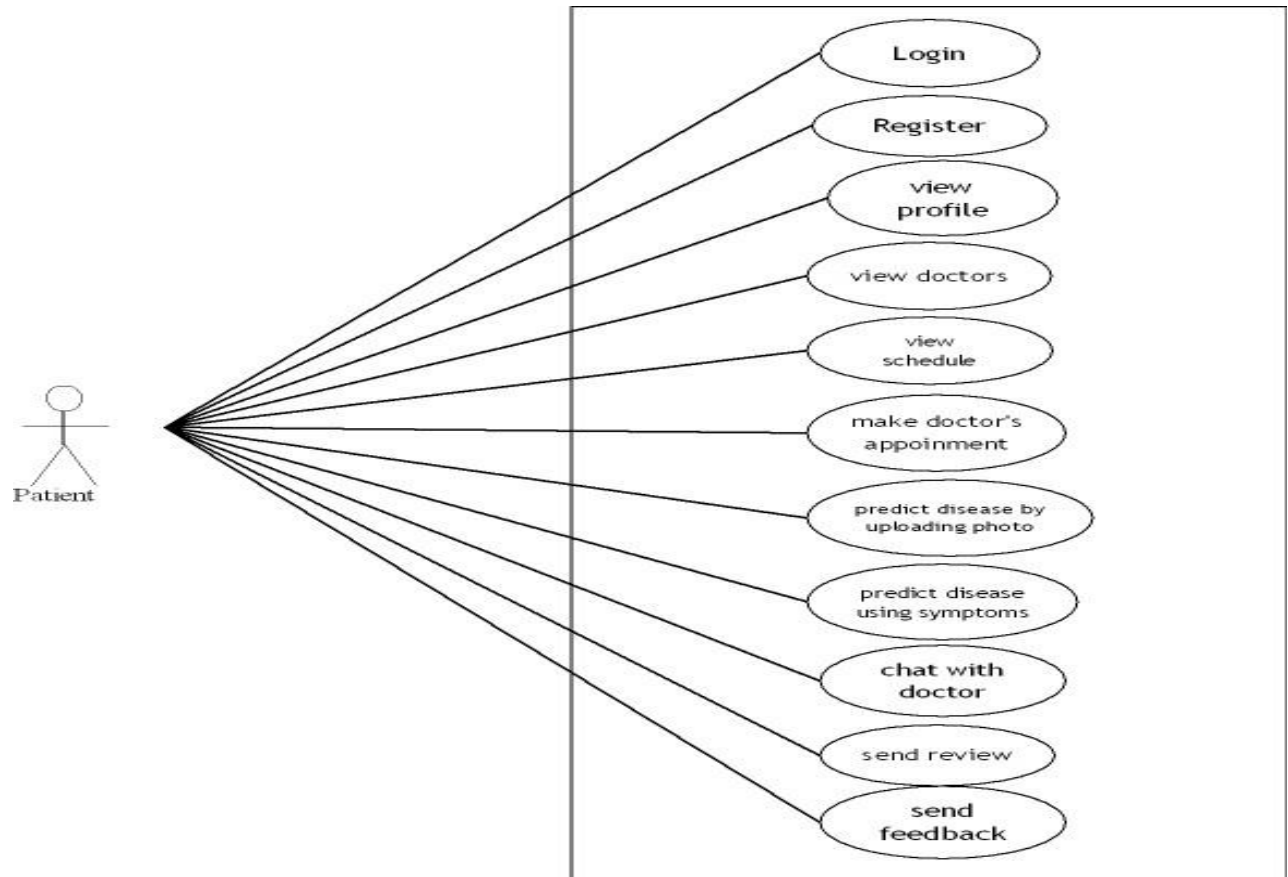
**Admin:**



**Doctor:**



## Patient:



### **3. SYSTEM DESIGN**

### **3.1 INTRODUCTION:**

System design provides an understanding of the procedural details, necessary implementing the system recommended in the feasibility study. Basically it is all about the creation of a new system. This is a critical phase since it decides the quality of the system and has a major impact on the testing and implementation phases.

**System design consists of three major steps.**

- Drawing of the expanded system data flow charts to identify all the processing functions required.
- The allocation of the equipment and the software to be used.
- The identification of the test requirements for the system.

### **CHARACTERS OF DESIGN**

- A design should exhibit a hierarchical organization that makes intelligent use of control among components of the software.
- A design should be modular that is, the software should be logical.
- A design should contain distinct and separable representation of data and procedure.
- A design should lead to interface that reduce the complexity of the connections between modules and with the external environment.

### **3.2 Database Design**

A Database is a collection of inter related data stored with minimum redundancy to serve many users quickly and efficiently. In database design data independence, accuracy, privacy and security are given higher priority. Database design is an integrated approach to the file design. This activity deals with the design of the physical data base. All entities and attributes have been identified while creating the database. The database design deals with the grouping of data into number of tables so as to.

- ✓ Reduplication of data.
- ✓ Minimize storage space.

- ✓ Retrieve the data efficiently.

Following are some guidelines for the database design:

- Design a relational schema so that it is easy to explain its meaning. Do not combine attributes from multiple entity and relationship type into a single relation.
- Design the database schema so that no insertion, deletion or modification anomalies are present in the relation.
- As far as possible, avoid placing attributes in the base relation whose values may frequently be null.
- Design relation schema so that they can be joined with equality conditions on attributes that are either primary keys or foreign keys in a way that no spurious tuples are generated.

### 3.3 Table Design

DB design is required to manage large bodies of information. The management of data involves both the definition of the structure of storage of information and provisions of mechanism for the manipulation of information. For developing an efficient database certain conditions have to be fulfilled such as:

- Control Redundancy
- Ease of Use
- Data Independence
- Accuracy and Integrity

There are five major steps in design process:

- Identify the table and relationship.
- Identify the data that is needed for each table and relationship.
- Resolve the relationship.
- Verify the design.
- Implement the design

The Database Consist of the following tables given below.



### 3.4 Ai Skin Expert

#### 1.Login table

Column name	Data type	Constraints	Description
login_id	int	primary key	unique identifier
username	varchar(50)	not null	name of user
password	varchar(50)	not null	secret key
type	varchar(50)	not null	To specify the role

#### 2. Appointment table

Colum name	Data type	Constraints	Description
appoint_id	int	primary key	unique identifier
user_id	int	Foreign key	
sched_id	int	Foreign key	Schedule id
token_no	int	not null	

#### 3. Chat table

Colum name	Data type	Constraints	Description
chat_id	int	primary key	unique identifier
date	date	not null	date
time	varchar(50)	not null	time
from_id	int	not null	
to_id	int	not null	
message	varchar(50)	not null	

4.disease table

Colum name	Data type	Constraints	Description
disease_id	int	primary key	unique identifier
disease_name	varchar(50)	not null	Name of the disease
description	varchar(50)	not null	About disease
dtype	varchar(50)	not null	
how_to_cure	varchar(50)	not null	

5.doctor table

Colum name	Data type	Constraints	Description
doctor_id	int	primary key	unique identifier
name	varchar(100)	not null	Name of the doctor
email	varcahar(100)	not null	
phone	varchar(100)	not null	
qualification	varchar(100)	not null	

6. Feedback table

Colum name	Data type	Constraints	Description
feedback_id	int	primary key	unique identifier
user_id	int	Primary key	
user_details	varchar(100)	not null	Details of the user
date	date	not null	
feedback	varchar(100)	not null	Feedback from user

### 7. Review table

Colum name	Data type	Constraints	Description
review_id	int	Primary key	unique identifier
date	varchar(50)	not null	
user_id	int	Foreign key	
doctor_id	int	Foreign key	
review	Varchar((100)	not null	

### 8. Schedule table

Colum name	Data type	Constraints	Description
shed_id	int	primary key	Unique identifier
Doctor_id	int	Foreign key	
date	varchar(100)	Not null	
Time_from	int	Not null	
Time_to	int	Not null	

### 9. Symptom table

Colum name	Data type	Constraints	Description
symptom_id	int	Not null	unique identifier
disease_id	int	Foreign key	
symptom_name	varchar(50)	Not null	

### 10. User table

Colum name	Data type	Constraints	Description
user_id	int	Primary key	login identifier
name	Varchar(20)	Not null	
email	Varchar(20)	Not null	
phone	int	Not null	
age	int	Not null	
gender	Varchar(50)	not null	

### 11. Bank table

Column name	Data type	Constraints	Description
bank_id	int	Primary key	Unique identifier
bank_name	Varchar(100)	Not null	
account_no	int	Not null	
ifsc_no	int	Not null	
balance	int	Not null	

### 12. Payment table

Column name	Data type	Constraints	Description
payment_id	int	Primary key	Unique identifier
appoint_id	int	Foreign key	
amount	int	Not null	

## 3.5. Data Flow Diagram

A graphical representation is used to describe and analyses the movement of data through a system manual or automated including the processes, Storing of data and delays in the system. Data flow diagrams are the central tool and the basis from which other components are developed.

The transformation of data, from input to output through process may be described logically and independently of the physical components associated with the system.

They are termed logical dataflow diagrams, showing the actual implementations and the movement of data between people, departments and

workstations. DFD is one of the most important modelling tools used in system design. DFD shows the flow of data through different process in the system.

**PURPOSE:**

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have reasonably completed model of the behavior of the system. It is important to avoid premature designs, wherein develop designs before analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

Throughout my project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In my opinion, "efficient design of the data flow and context flow diagram helps to design the system successfully without much major flaws within the scheduled time". This is the most complicated part in a project. In the designing process, my project took more than the activities in the software lifecycle. If we design a system efficiently with all the future enhancements the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as a way of expressing system requirements in graphical form. A data flow diagram also known as "bubble chare" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It functionality decomposes the requirement specification down to the lowest level. Data Flow Diagram depicts the

information flow, the transformation flow and the transformations that are applied as data move from input to output. Thus DFD describes what data flows rather than how they are processed.

Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is one of the most important tools used during system analysis. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

Data Flow Diagrams are made up of a number of symbols, which represents system components. Data flow modelling method uses four kinds of symbols, which are used to represent four kinds of system components.

These are

- Process
- Data stores
- Data flows
- External entity

#### **Process:**

Process shows the work of the system. Each process has one or more data inputs and produce one or more data outputs. Processes are represented by rounded rectangles in Data Flow Diagram. Each process has a unique name and number. This name and number appears inside the rectangle that represents the process in a Data Flow Diagram.

#### **Data Stores:**

A data stores is a repository of data. Processes can enter data, into a store or retrieve the data from the data store. Each data has a unique name.

#### **Data Flows:**

Data flows show the passage of data in the system and are represented by lines joining system components. An arrow indicates the direction of flow and the line is labelled by name of the dataflow.

#### **External Entity:**

External entities are outside the system but they either supply input data into the system or use other systems output. They are entities on which the designer has control. They may be an organizations customer or other bodies with which the system interacts. External entities that supply data into the system are sometimes called source. External entities that use the system data are sometimes called sinks. These are represented by rectangles in the

## Data Flow Diagram.

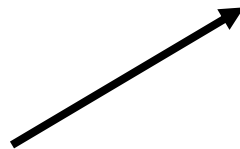
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Basic data flow diagram symbols are.....

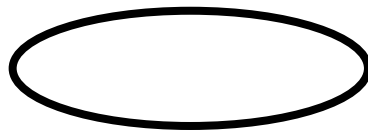
- A Square defines a source (originator) or destination of a system data:



- An Arrow identifies data flow. It is a pipeline through which information flows:



- A Circle represents a process that transforms incoming data flow(s) into outgoing data flow(s):



- An Open Rectangle is a data store:

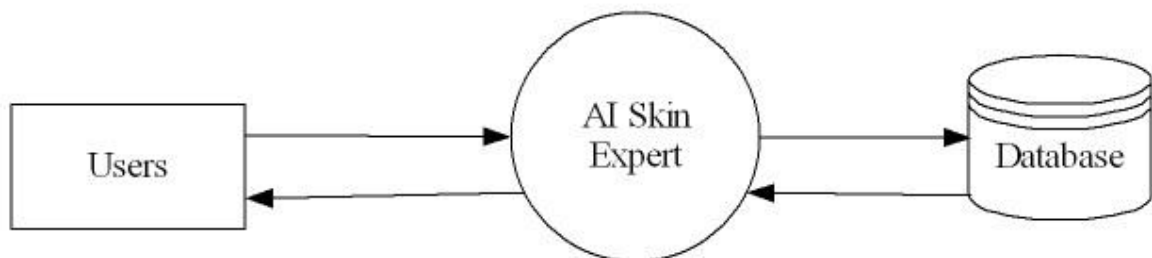


**Four steps are commonly used to construct a DFD:**

- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and left to right.
- When a process is exploded in to lower level details they are numbered.
- The names of data stores, sources and destinations are written in Capital letters.

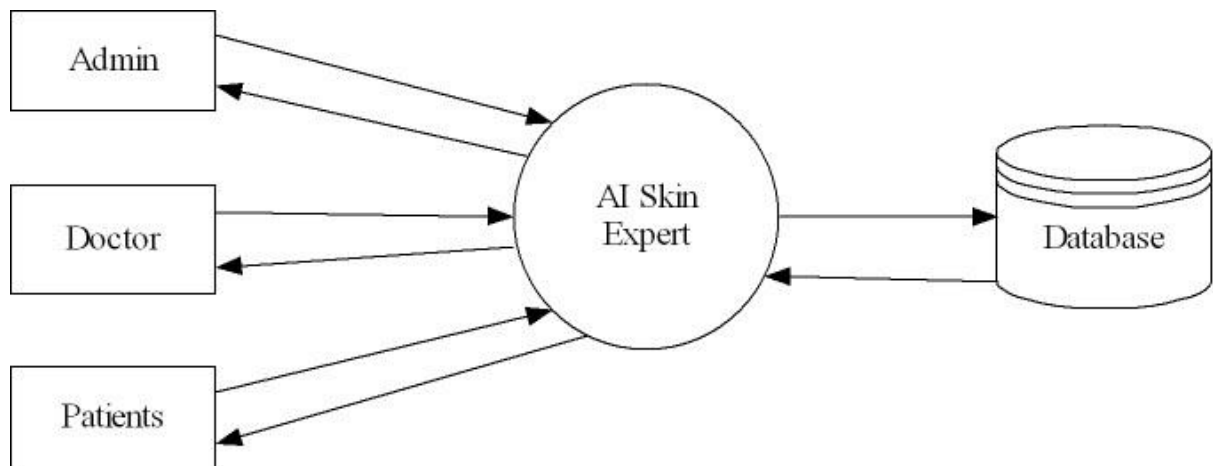
**DFD Level-0**

**Level 0**

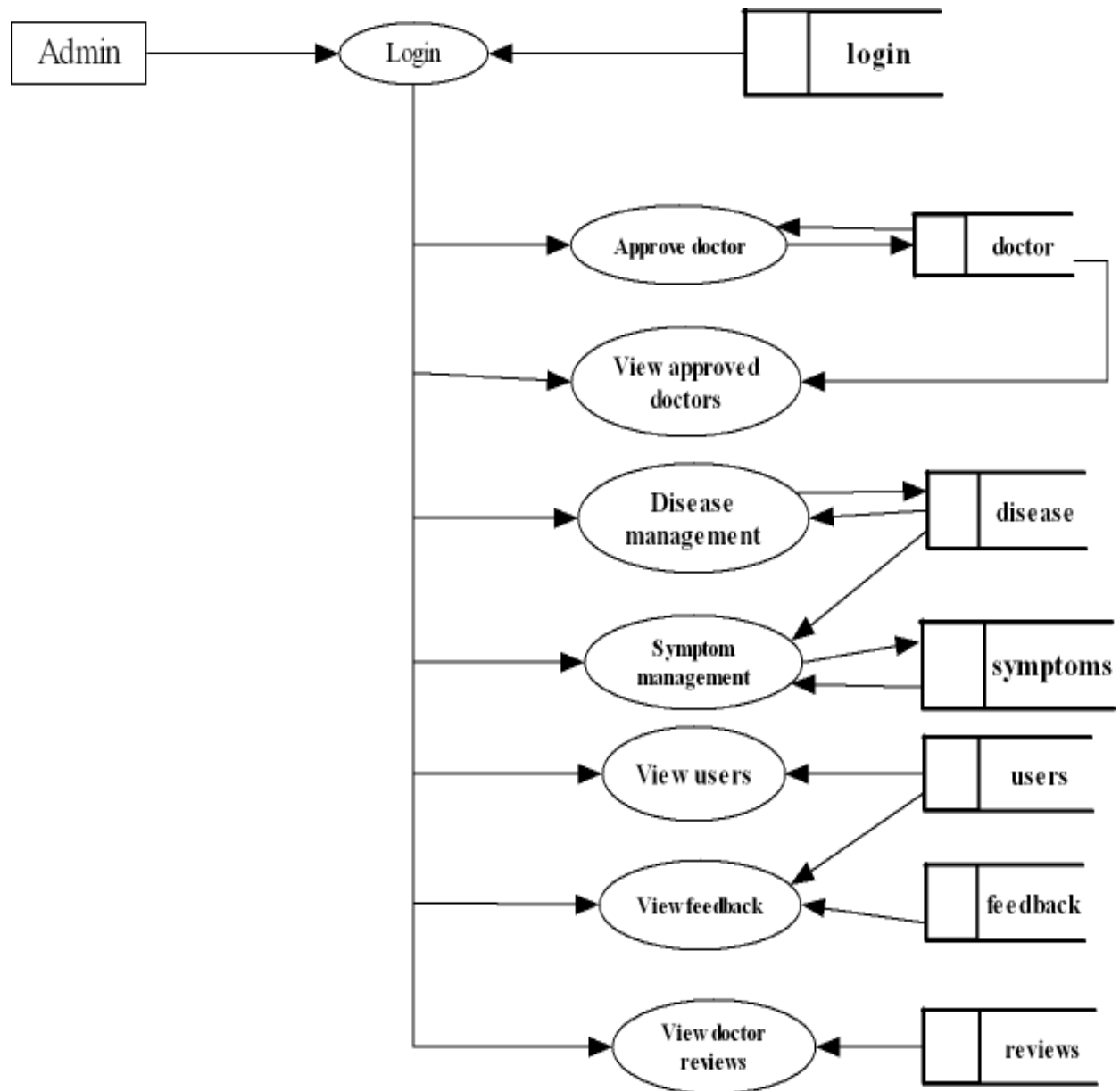




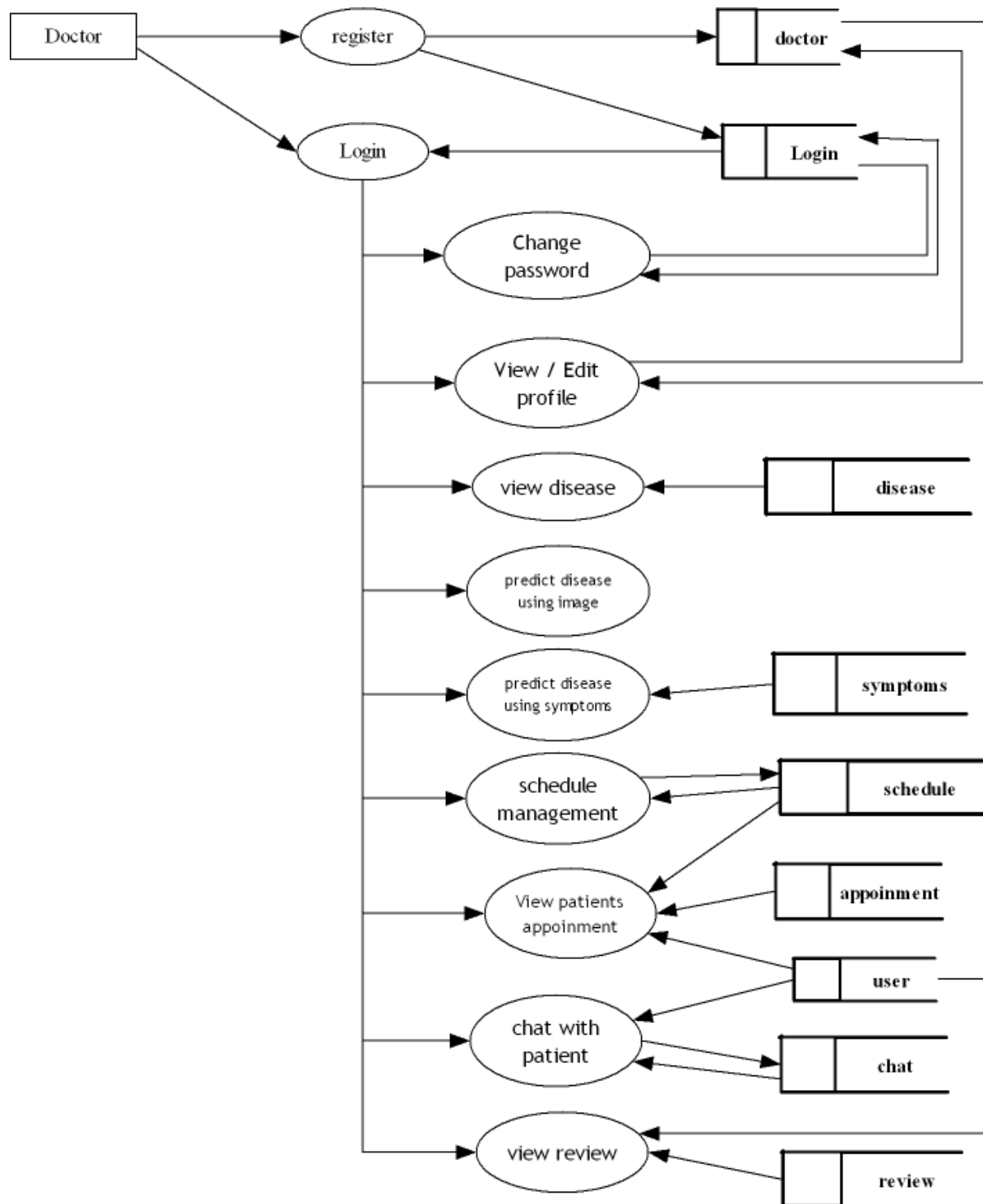
## DFD Level-1



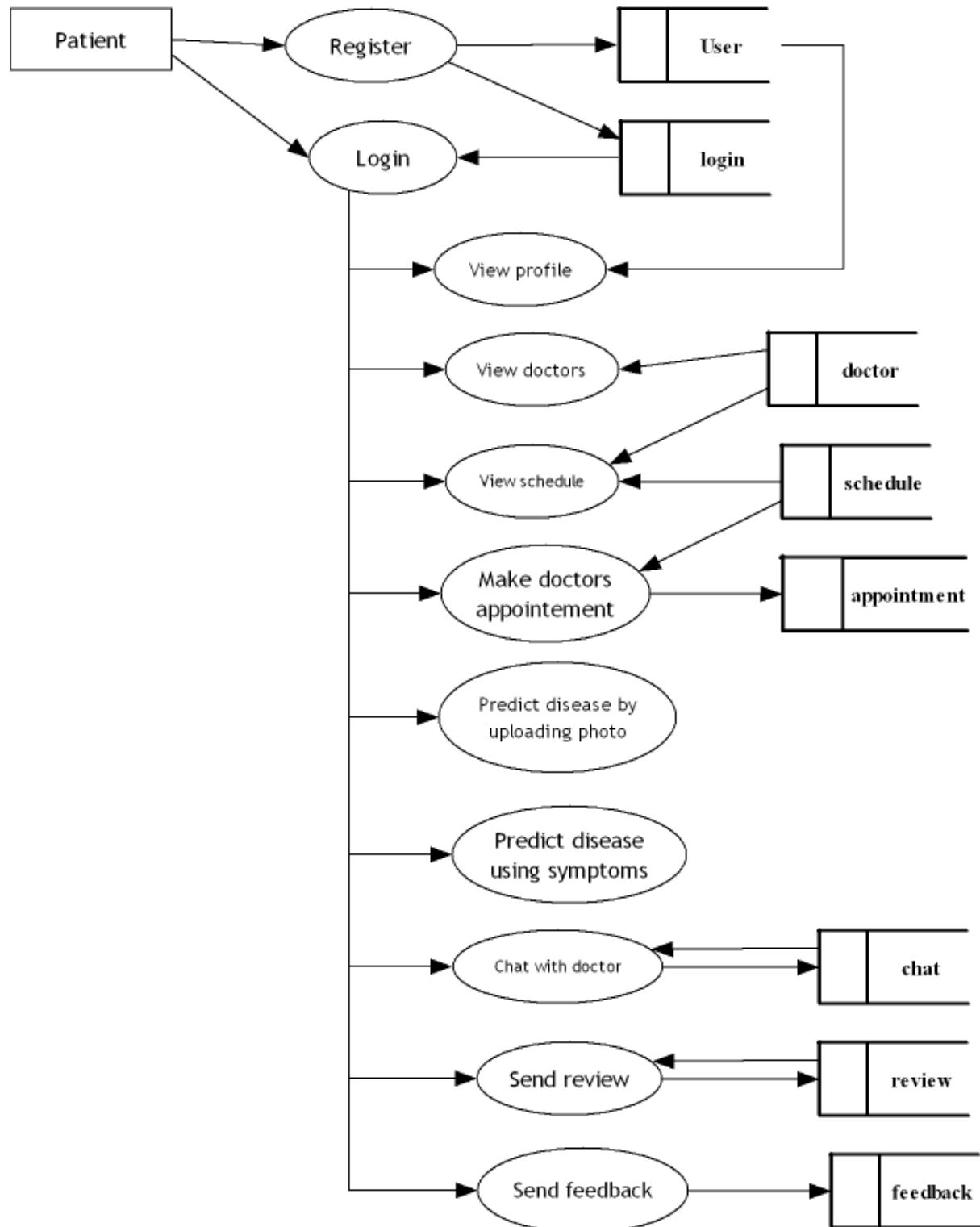
## DFD Level-1.1 Admin



## DFD Level-1.2 Doctor



### DFD Level-1.3 STUDENT



### 3.6 ER Diagram

An ER diagram is a diagram that helps to design databases in an efficient way. It is a data model for describing the data or information. It is a visual representation of data that describes how data is related to each other. The main components of ER models are entities, attributes and the relationships that can exist among them.

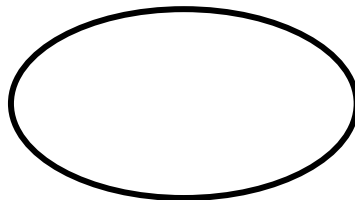
#### Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.



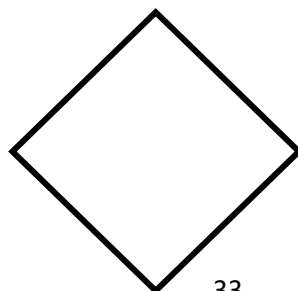
#### Attribute

Attributes are properties of entities. Attributes are represented by means of ellipses. Every ellipse represents one attribute and is directly connected to its entity (rectangle).

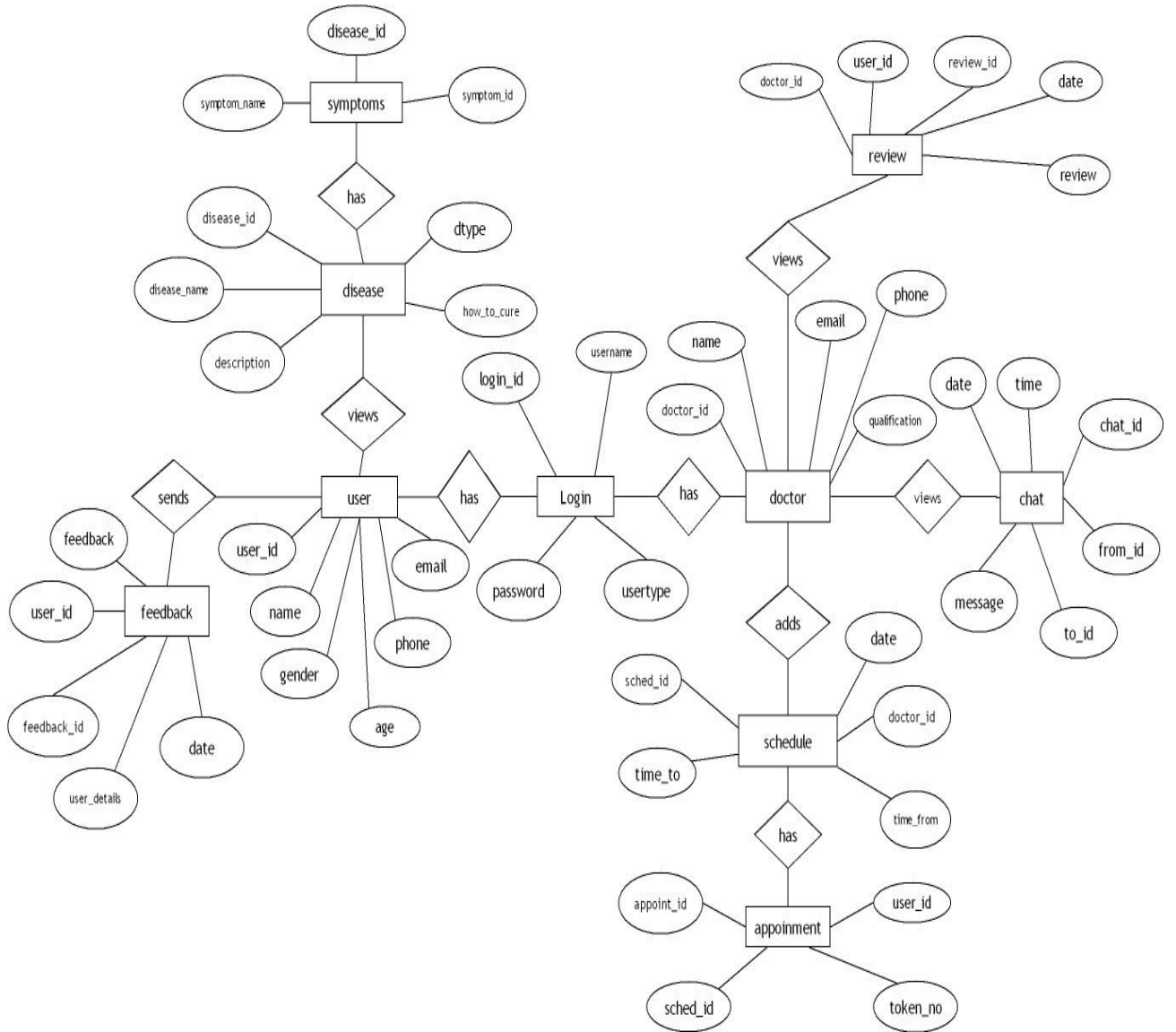


#### Relationship

Relationships are represented by diamond shaped box. Name of the relationship is written in the diamond box. All entities (rectangles), participating in relationship, are connected to it by a line.



## Architectural design



## **4. CODING**

## **4.1 INPUT INTERFACE**

Input design is a part of overall system design, which requires very careful attention. If data going into the system is correct, then the processing and output will magnify these errors. Thus the designer has a number of clear objectives in the different stages of input design.

- To produce a cost effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that input is acceptable to and understand by the user.

## **4.2 OUTPUT INTERFACE**

At the beginning of the output design various types of outputs such as external, internal, operational and interactive and turn around are defined. Then the format, content, location, frequency, volume and sequence of the outputs are specified. The content of the output must be defined in detail. The system analysis has two specific objectives at this stage.

- To interpret and communicate the results of the computer part of a system to the users in a form, which they can understand, and which meets their requirements.
- To communicate the output design specifications to programmers in a way in which it is unambiguous, comprehensive and capable of being translated into a programming language.

## **4.3 TECHNOLOGY SPECIFICATION**

The proposed system is developed using HTML , CSS, JavaScript for web and Java(Android Studio) for Android as Front end. MySQL ,flask and python for web and Python ,MySQL for Android as Back end.

### **1. FRONT END**

#### **HTML**

HTML is an acronym which stands for Hyper Text Markup Language which is used for creating web pages and web applications. Let's see what is meant by Hypertext Markup Language, and Web page. Hyper Text: HyperText simply means "Text within Text." A text has a link within it, is a hypertext. Whenever you click on a link which brings you to a new webpage, you have clicked on a hypertext. HyperText is a way to link two or more web pages



(HTML documents) with each other. Markup language: A markup language is a computer language that is used to apply layout and formatting conventions to a text document. Markup language makes text more interactive and dynamic. It can turn text into images, tables, links, etc. Web Page: A web page is a document which is commonly written in HTML and translated by a web browser. A web page can be identified by entering an URL. A Web page can be of the static or dynamic type. With the help of HTML only, we can create static web pages. Hence, HTML is a markup language which is used for creating attractive web pages with the help of styling, and which looks in a nice format on a web browser. An HTML document is made of many HTML tags and each HTML tag contains different content

## **Features**

- 1) It is a very easy and simple language. It can be easily understood and modified.
- 2) It is very easy to make an effective presentation with HTML because it has a lot of formatting tags.
- 3) It is a markup language, so it provides a flexible way to design web pages along with the text.
- 4) It facilitates programmers to add a link on the web pages (by html anchor tag), so it enhances the interest of browsing of the user.
- 5) It is platform-independent because it can be displayed on any platform like Windows, Linux, and Macintosh, etc.
- 6) It facilitates the programmer to add Graphics, Videos, and Sound to the web pages which makes it more attractive and interactive.
- 7) HTML is a case-insensitive language, which means we can use tags either in lowercase or upper-case.

## **HTML Versions**

Since the time HTML was invented there are lots of HTML versions in market, the brief introduction about the HTML version is given below:

**HTML 1.0:** The first version of HTML was 1.0, which was the barebones version of HTML language, and it was released in 1991.

**HTML 2.0:** This was the next version which was released in 1995, and it was standard language version for website design. HTML 2.0 was able to support extra features such as form-based file upload, form elements such as text box, option button, etc.

**HTML 3.2:** HTML 3.2 version was published by W3C in early 1997. This version was capable of creating tables and providing support for extra options for form elements. It can also support a web page with complex mathematical equations. It became an official standard for any browser till January 1997. Today it is practically supported by most of the browsers.

**HTML 4.01:** HTML 4.01 version was released on December 1999, and it is a very stable version of HTML language. This version is the current official standard, and it provides added support for stylesheets (CSS) and scripting ability for various multimedia elements.

**HTML5** : HTML5 is the newest version of Hyper Text Markup language. The first draft of this version was announced in January 2008. There are two major organizations one is W3C (World Wide Web Consortium), and another one is WHATWG( Web Hypertext Application Technology Working Group) which are involved in the development of HTML 5 version, and still, it is under development.

### **Description of HTML example**

**<!DOCTYPE>**: It defines the document type or it instruct the browser about the version of HTML

**<html>** :This tag informs the browser that it is an HTML document. Text between html tag describes the web document. It is a container for all other elements of HTML except<!DOCTYPE>

**<head>** : It should be the first element inside the element, which contains the metadata(information about the document). It must be closed before the body tag opens.

**<title>**: As its name suggests it is used to add title of that html page which appears at the top of the browser window. It must be placed inside the head tag and should close immediately (optional).

**<body>** : Text between body tag describes the body content of the page that is visible to the end user . This tag contains the main content of the html document.

**<h1>** : Text between <h1> tag describes the first level heading of the webpage.

**<p>**: Text between <p> tag describes the paragraph of the webpage.

### **CSS (Cascading Style Sheet)**

CSS is used to control the style of a web document in a simple and easy way.CSS is the acronym for "**Cascading Style Sheet**".**Cascading Style Sheets (CSS)** is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript .

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

## History of CSS

CSS was first proposed by **HakonWium Lie** on October 10, 1994. At the time, Lie was working with Tim Berners-Lee (father of Html) at CERN. The European Organization for Nuclear Research is known as CERN. Hakonwium lie is known as father of css.CSS was proposed in 1994 as a web styling language, to solve some of the problems of Html 4. There were other styling languages proposed at this time, such as Style Sheets for Html and JSSS but CSS won.

## Why to learn CSS?

**Cascading Style Sheets**, fondly referred to as **CSS**, is a simple design language intended to simplify the process of making web pages presentable.<sup>23</sup> CSS is a **MUST** for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain.

I will list down some of the key advantages of learning CSS:

- **Create Stunning Web site** - CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.
- **Become a web designer** - If you want to start a carrier as a professional web designer, HTML and CSS designing is a must skill.
- **Control web** - CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.
- **Learn other languages** - Once you understand the basic of HTML and CSS then other related technologies like javascript, php, or angular are become easier to understand.

## Types of CSS

There are three ways of inserting a style sheet in any Html documents, they are given below:

- Inline style sheet
- Internal style sheet
- External style sheet

Inline CSS is use with any elements of HTML where it is used on page. Here we use inline CSS for paragraph, the example shows how to change the color and the left margin of a paragraph. An internal style sheet should be used when a single document has a unique style. Internal styles sheet is defined in the head section of an HTML page, by using the <style> tag. An external style sheet is ideal when the style is applied to many pages. With an external style sheet, we can change the look of an entire Web site by changing one file.

## CSS Selectors

Selectors are used for select an Html element it is select by name, id, class etc.

1. id selector
2. class selector
3. Element Selector
4. Group Selector
5. Universal Selector

## Features

1. A style rule consists of a selector component and a declaration block component.
2. The selector is used to point to the HTML component which you want to get styled.
3. Inside the declaration block, one or more declarations are contained along with semicolons.
4. Every declaration which is put has a CSS property name, a semicolon, and a value. For example, color is the property and the value is red in color. Font size is the property and the 15px is the value.
5. CSS declaration ends with a semicolon and these blocks are surrounded by curly braces.
6. CSS selectors are the ones which are used to find HTML elements which are based on the element name, id, attribute, class and more.
7. One unique element will be selected by the ID of an element.
8. If you wish to select the particular element with a specific id, the # function along with the id attribute should be used.
9. If you wish to select the elements with a specific class, the period character along with the name class should be written.
10. Universal selector: If you are not interested in choosing the elements of a certain type, the universal selector simply matches with the element name.
11. Element selector: These selectors choose the element based on the element name.
12. Descendent selector: When a particular element lies inside another element, then it is called as the descendent selector.
13. ID selector: This selector uses the id of the HTML element so that a specific element could be selected.
14. Class selectors: It selects the element with a specific class attribute.
15. Grouping selectors: It will be a good option to group the selectors so as to minimize the code. Each selector along with a comma should be used to group the selectors.

## JavaScript

**JavaScript** is a object-based scripting language and it is light weighted. It is first implemented by Netscape (with help from Sun Microsystems). JavaScript was created by Brendan Eich at Netscape in 1995 for the purpose of allowing code in web-pages (performing logical operation on client side).It is not compiled but translated. JavaScript Translator is responsible to translate the JavaScript code which is embedded in

browser. Netscape first introduced a JavaScript interpreter in Navigator 2. The interpreter was an extra software component in the browser that was capable of interpreting JavaScript source code inside an HTML document. This means that web page developers do not need other software other than a text editor to develop any web page.

### **Why we use JavaScript?**

Using HTML we can only design a web page but you cannot run any logic on web browser like addition of two numbers, check any condition, looping statements (for, while), decision making statement (if-else) at client side. All these are not possible using HTML so to perform all these tasks at client side you need to use JavaScript.

### **History**

JavaScript is an object-based scripting language and it is light weighted. It is first implemented by Netscape (with help from Sun Microsystems). JavaScript was created by Brendan Eich at Netscape in 1995 for the purpose of allowing code in web-pages (performing logical operation on client side). Using HTML we can only design a web page but you cannot run any logic on web browser like addition of two numbers, check any condition, looping statements (for, while), decision making statement (if-else) etc. All these are not possible using HTML so to perform all these tasks, we use JavaScript. Using HTML we can only design a web page if we want to run any programs like C programming we use JavaScript. Suppose we want to print sum of two numbers then we use JavaScript for coding.

### **Features**

- JavaScript is an object-based scripting language.
- Giving the user more control over the browser.
- It handles dates and time.
- It detects the user's browser and OS.
- It is light weighted.
- JavaScript is a scripting language and it is not Java.
- JavaScript is an interpreter-based scripting language.
- JavaScript is case sensitive.
- JavaScript is an object-based language as it provides predefined objects.
- Every statement in JavaScript must be terminated with a semicolon (;).
- Most of the JavaScript control statements syntax is the same as the syntax of control statements in C language. An important part of JavaScript is the ability to create new functions within scripts. Declare a function in JavaScript using the function keyword.

### **Android**

**Android** is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming

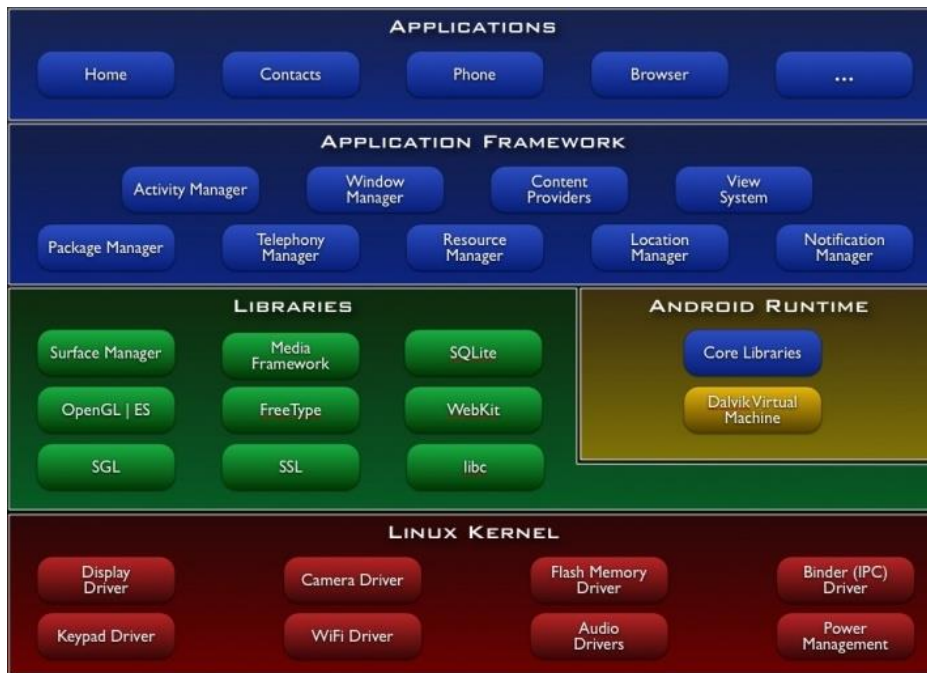
language. Android applications are written in the Java programming language. The Android SDK tools compile the code—along with any data and resource files— into an Android package, an archive file with an .apk suffix. All the code in a single .apk file is considered to be one application and is the file that Android-powered devices use to install the application. Application components are the essential building blocks of an Android application. Each component is a different point through which the system can enter your application. Not all components are actual entry points for the user and some depend on each other, but each one exists as its own entity and plays a specific role—each one is a unique building block that helps define application's overall behavior.

## **Features**

- Application framework enabling reuse and replacement of components
- Dalvik virtual machine optimized for mobile devices
- Integrated browser based on the open source Web Kit engine
- Optimized graphics powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
- Media support for common audio, video, and still image formats (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
- GSM Telephony (hardware dependent)
- Bluetooth, EDGE, 3G, and Wi-Fi (hardware dependent)
- Camera, GPS, compass, and accelerometer (hardware dependent)
- Rich development environment including a device emulator, tools for debugging, memory and performance profiling, and a plug-in for the Eclipse IDE.

## **ANDROID ARCHITECTURE**

The following diagram shows the major components of the Android operating system. Each section is described in more detail below.



## APPLICATION FRAMEWORK

By providing an open development platform, Android offers developers the ability to build extremely rich and innovative applications. Developers are free to take advantage of the device hardware, access location information, run background services, set alarms, add notifications to the status bar, and much, much more. Developers have full access to the same framework APIs used by the core applications. The application architecture is designed to simplify the reuse of components; any application can publish its capabilities and any other application may then make use of those capabilities (subject to security constraints enforced by the framework). This same mechanism allows components to be replaced by the user.

Underlying all applications is a set of services and systems, including:

A rich and extensible set of the views that can be used to build an application, including lists, grids, text boxes, buttons, and even an embeddable web browser

Content Providers that enable applications to access data from other applications (such as Contacts), or to share their own data

A Resource Manager, providing access to non-code resources such as localized strings, graphics, and layout files

A Notification Manager that enables all applications to display custom alerts in the status bar

An Activity Manager that manages the lifecycle of applications and provides a common navigation back stack.

## Libraries

Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below:

- System C library - a BSD-derived implementation of the standard C system library (libc), tuned for embedded Linux-based devices
- Media Libraries - based on Packet Video's Open CORE; the libraries support playback and recording of many popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG
- Surface Manager - manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications
- LibWebCore - a modern web browser engine which powers both the Android browser and an embeddable web view
- SGL - the underlying 2D graphics engine
- 3D libraries - an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer
- Free Type - bitmap and vector font rendering

## **Android Runtime**

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language. Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple VMs efficiently. The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been transformed into the .dex format by the included "dx" tool. The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management. Linux Kernel Android relies on Linux version 2.6 for core system services such as security, memory management, process management, network stack, and driver model. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack.

## **Activity Lifecycle**

Activities in the system are managed as an activity stack. When a new activity is started, it is placed on the top of the stack and becomes the running activity -- the previous activity always remains below it in the stack, and will not come to the foreground again until the new activity exits.

An activity has essentially four states:

- If an activity is in the foreground of the screen (at the top of the stack), it is active or running.
- If an activity has lost focus but is still visible (that is, a new non-full sized or transparent activity has focus on top of your activity), it is paused. A paused activity is completely alive (it maintains all state and member information and remains attached to the window manager), but can be killed by the system in extreme low memory situations.



- If an activity is completely obscured by another activity, it is stopped. It still retains all state and member information, however, it is no longer visible to the user so its window is hidden and it will often be killed by the system when memory is needed elsewhere.
- If an activity is paused or stopped, the system can drop the activity from memory by either asking it to finish, or simply killing its process. When it is displayed again to the user, it must be completely restarted and restored to its previous state.

## **Android Studio**

**Android Studio** is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (ADT) as primary IDE for native Android application development. Android Studio supports all the same programming languages of IntelliJ, and PyCharm and Android Studio 3.0 supports Java 7 language features and a subset of Java 8 language features that vary by platform version. Features like Gradle-based build support, Android-specific refactoring and quick fixes, a rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations, Android Virtual Device (Emulator) to run and debug apps in the Android studio, etc. are provided in the current stable version.

## **About JAVA**

The first version of Java began in 1991 and was written in 18 months at Sun micro system. In fact, it wasn't even called Java in those days it was Oak, and it was used internally at sun.

Java had adopted a model that made it perfect for the Internet, the byte code model. It is implemented as the Java virtual Machine (JVM), which is the application that actually runs the java program. When JVM is installed on a computer, it can run java programs. Java programs, before, don't need to be self-sufficient, and they don't have to include all the machine-level code that actually runs on the computer. In this way, our Java program can be very small, because all the machine-level code to run our program is already on the target computer and doesn't have to be downloaded.

When it executes a program, the JVM can strictly monitor what goes on, which makes it great for Internet Applications.

## **JAVA FEATURES**

The inventors of java wanted to design a language, which could offer solutions to some of the problems encountered in modern programming. They wanted the language to be reliable, portable and distributed but also simple, compact and interactive. Sun Microsystems officially describes java with the following attributes.

- Compiled and Interpreted
- Platform-Independent and Portable
- Object-Oriented
- Robust and Secure

- Distributed
- Familiar, Simple and Small
- Multithreaded and Interactive
- High Performance
- Dynamic and Extensible

## **Compiled and Interpreted**

Usually a computer language is either compiled or interpreted. Java combines both these approaches thus making java a two-stage system first java compiler translate source code in to byte code instructions. Byte-codes are not machine code that can be directly executed by the machine that is running the java program. Platform Independent and Portable.

The most significant contribution java over other languages is its portability. Java programs can be easily moved from one computer system to another, anywhere at any time. Changes and Upgrades in 27 operating systems, processors and system resources will not force any changes in java programs. This is the reason why java has become a popular language for programming on internet, which interconnects different kinds of systems worldwide. Java ensures portability in two ways. First, java compiler generates byte code instructions that can implemented on any machine. Secondly, the sizes of the primitive data types are machine independent.

## **Object-Oriented**

Java is a true Object-Oriented Language. Almost everything in Java is an Object. All program code and data reside within objects and classes. Java comes with an extensive set of classes arranged in packages that we can use in our programs by inheritance. The object model in java is simple and easy to extend.

## **Robust and Secure**

Java is a robust language. It provides many safe guards to ensure reliable code. It has strict compile time checking for data types. It is designed as garbage collected language. Java also incorporates with the concept of exception handling.

## **Distributed**

Java is designed as a distributed language for creating application on network. It has the ability to share both data & Program.

## **Multithreaded and interactive**

Multithreaded means handling multiple tasks simultaneously java supports multithreaded programs. This means that we not wait for the application to finish one task before beginning another.

## **High performance**

Java performance is impressive for an interpreted language mainly due to the use of intermediate byte code. Java architecture is also designed to reduce overheads during runtime.

## **2 BACK END**

### **Python**

Python is an object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

### **MySQL**

MySQL software is Open Source

Open source means that it is possible for anyone to use and modify. Anybody can download the MySQL software from the Internet and use it without paying anything. The MySQL database server is very fast, reliable, and easy to use. It was originally developed

to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Though under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet

- An object-oriented interface
- Support for prepared statements
- Support for multiple statements
- Support for transactions
- Enhanced debugging support
- Embedded server support

### **Pycharm:**

**PyCharm** is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains (formerly known as IntelliJ).<sup>[5]</sup> It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as data science with Anaconda.<sup>[6]</sup>

#### **FEATURES**

- Coding assistance and analysis, with code completion, syntax and error highlighting, linter integration, and quick fixes
- Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
- Python refactoring: includes rename, extract method, introduce variable, introduce constant, pull up, push down and others
- Integrated Python debugger
- Integrated unit testing, with line-by-line code coverage
- Version control integration: unified user interface for Mercurial, Git, Subversion, Perforce and CVS with change lists and merge

### **Flask**

**Flask** is a micro web framework written in Python. It is classified as microframework because it does not require particular tools or libraries.<sup>[2]</sup> It has no

database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

## **15.CODING PAGES**

## 5.1 SOURCE CODE

### 1. AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.lenovo.aiskinexpert">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/icon"
        android:label="@string/app_name"
        android:roundIcon="@drawable/icon"
        android:supportRtl="true"
        android:theme="@style/AppTheme"
        android:usesCleartextTraffic="true">
        <uses-library
            android:name="org.apache.http.legacy"
            android:required="false" />

        <activity android:name=".ip_page">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".patient__signup" />
        <activity android:name=".login" />
        <activity android:name=".view_profile" />
        <activity android:name=".view_doctor" />
        <activity android:name=".Send_feedback" />
        <activity android:name=".send_review" />
        <activity android:name=".view_schedule" />
        <activity android:name=".view_booking" />
        <activity android:name=".Chat" />
        <activity
            android:name=".home_page"
            android:label="@string/title_activity_home_page"
```

```

        android:theme="@style/AppTheme.NoActionBar" />
        <activity android:name=".costum_view_doctor" />
        <activity android:name=".custom_view_schedule" />
        <activity android:name=".custom_view_booking" />
        <activity android:name=".payment" />
        <activity android:name=".prediction_image" />
        <activity android:name=".predict_by_symptom"></activity>
    </application>

</manifest>

```

## 2.Login.java

```

package com.example.lenovo.aiskinexpert;

import android.content.Intent;
import android.content.SharedPreferences;
import android.preference.PreferenceManager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.android.volley.DefaultRetryPolicy;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONArray;
import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;

public class login extends AppCompatActivity implements View.OnClickListener {
    EditText ed_name;
    EditText ed_pass;
    Button b1;
    TextView t1;

    @Override

```



```

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_login);

    ed_name=findViewById(R.id.editText3);
    ed_pass=findViewById(R.id.editText4);
    b1=findViewById(R.id.button3);
    t1=findViewById(R.id.textView);
    b1.setOnClickListener(this);
    t1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent ij=new Intent(getApplicationContext(), patient__signup.class);
            startActivity(ij);
        }
    });
}

@Override
public void onClick(View view) {
    final String username=ed_name.getText().toString();
    final String password=ed_pass.getText().toString();

    final SharedPreferences sh=
PreferenceManager.getDefaultSharedPreferences(getApplicationContext());
    String hu = sh.getString("url", "");
    String url = hu + "/and_login";

    RequestQueue requestQueue = Volley.newRequestQueue(getApplicationContext());
    StringRequest postRequest = new StringRequest(Request.Method.POST, url,
        new Response.Listener<String>() {
        @Override
        public void onResponse(String response) {
            // Toast.makeText(getApplicationContext(), response,
Toast.LENGTH_LONG).show();

            try {
                JSONObject jsonObj = new JSONObject(response);
                if (jsonObj.getString("status").equalsIgnoreCase("ok")) {
//                    Toast.makeText(Login.this, "welcome",
Toast.LENGTH_SHORT).show();
                    String typ = jsonObj.getString("type");
                    String id = jsonObj.getString("lid");
                    SharedPreferences.Editor ed =sh.edit();
                    ed.putString("lid", id);

```

```

        ed.commit();
        if (typ.equalsIgnoreCase("user")) {
            Toast.makeText(getApplicationContext(), "Welcome",
Toast.LENGTH_LONG).show();
            Intent i = new Intent(getApplicationContext(), home_page.class);
            startActivity(i);
        }
        } else {
            Toast.makeText(getApplicationContext(), "Not found",
Toast.LENGTH_LONG).show();
        }

    } catch (Exception e) {
        Toast.makeText(getApplicationContext(), "Error" +
e.getMessage().toString(), Toast.LENGTH_SHORT).show();
    }
}

},
new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        // error
        Toast.makeText(getApplicationContext(), "eeeeee" + error.toString(),
Toast.LENGTH_SHORT).show();
    }
}
) {
    @Override
    protected Map<String, String> getParams() {
        SharedPreferences sh =
PreferenceManager.getDefaultSharedPreferences(getApplicationContext());
        Map<String, String> params = new HashMap<String, String>();

        params.put("usr", username);
        params.put("psw", password);

        return params;
    }
};

int MY_SOCKET_TIMEOUT_MS=100000;

postRequest.setRetryPolicy(new DefaultRetryPolicy(
    MY_SOCKET_TIMEOUT_MS,
    DefaultRetryPolicy.DEFAULT_MAX_RETRIES,
    DefaultRetryPolicy.DEFAULT_BACKOFF_MULT));
requestQueue.add(postRequest);

}
}

```

### 3.Patient\_signup.java

```
package com.example.lenovo.aiskinexpert;

import android.content.Intent;
import android.content.SharedPreferences;
import android.preference.PreferenceManager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.Toast;

import com.android.volley.DefaultRetryPolicy;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;

public class patient__signup extends AppCompatActivity implements
View.OnClickListener {
    EditText ed_name;
    EditText ed_mail;
    EditText ed_phone;
    EditText ed_age;
    RadioButton r1;
    RadioButton r2;
    EditText ed_pass;
    Button b;
    String gender="";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_patient__signup);
```

```

    ed_name=findViewById(R.id.editText2);
    ed_mail=findViewById(R.id.editText5);
    ed_phone=findViewById(R.id.editText6);
    ed_age=findViewById(R.id.editText7);
    r1=findViewById(R.id.radioButton);
    r2=findViewById(R.id.radioButton2);
    ed_pass=findViewById(R.id.editText8);
    b=findViewById(R.id.button2);
    b.setOnClickListener(this);
}

@Override
public void onClick(View view) {
    final String ed_nm=ed_name.getText().toString();
    final String ed_ml=ed_mail.getText().toString();
    final String ed_ph=ed_phone.getText().toString();
    final String ed_ag=ed_age.getText().toString();
    final String ed_pss=ed_pass.getText().toString();
    String pattern="[a-zA-Z0-9._-]+@[a-z]+\\.[a-z]+";

    int flag=0;
    if(ed_nm.equalsIgnoreCase("")){
        ed_name.setError("required");
        flag++;
    }
    if(ed_ml.equalsIgnoreCase("")){
        ed_mail.setError("required");
        flag++;
    }
    if(ed_ph.equalsIgnoreCase("")){
        ed_phone.setError("required");
        flag++;
    }
    if(ed_ag.equalsIgnoreCase("")){
        ed_age.setError("required");
        flag++;
    }
    if(ed_pss.equalsIgnoreCase("")){
        ed_pass.setError("required");
        flag++;
    }
    if(ed_nm.equalsIgnoreCase("")){
        ed_name.setError("required");
        flag++;
    }
}

```

```

if(flag==0) {

    if (r1.isChecked()) {
        gender = "Male";
    }
    if (r2.isChecked()) {
        gender = "Female";
    }

    SharedPreferences sh =
PreferenceManager.getDefaultSharedPreferences(getApplicationContext());
    String hu = sh.getString("url", "");
    String url = hu + "/and_register";

    RequestQueue requestQueue = Volley.newRequestQueue(getApplicationContext());
    StringRequest postRequest = new StringRequest(Request.Method.POST, url,
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                // Toast.makeText(getApplicationContext(), response,
Toast.LENGTH_LONG).show();

                // response
                try {
                    JSONObject jsonObj = new JSONObject(response);
                    if (jsonObj.getString("status").equalsIgnoreCase("ok")) {
                        Toast.makeText(getApplicationContext(), "Registered",
Toast.LENGTH_LONG).show();
                        Intent ij = new Intent(getApplicationContext(), login.class);
                        startActivity(ij);
                    }

                    // }
                    else {
                        Toast.makeText(getApplicationContext(), "Invalid details",
Toast.LENGTH_LONG).show();
                    }

                } catch (Exception e) {
                    Toast.makeText(getApplicationContext(), "Error" +
e.getMessage().toString(), Toast.LENGTH_SHORT).show();
                }
            }
        },
        new Response.ErrorListener() {
            @Override

```

```

        public void onErrorResponse(VolleyError error) {
            // error
            Toast.makeText(getApplicationContext(), "eeee" + error.toString(),
Toast.LENGTH_SHORT).show();
        }
    }
} {
    @Override
    protected Map<String, String> getParams() {
        SharedPreferences sh =
PreferenceManager.getDefaultSharedPreferences(getApplicationContext());
        Map<String, String> params = new HashMap<String, String>();

        params.put("nam", ed_nm);
        params.put("em", ed_ml);
        params.put("ph", ed_ph);
        params.put("ag", ed_ag);
        params.put("gnd", gender);
        params.put("psw", ed_pss);

        return params;
    }
};

int MY_SOCKET_TIMEOUT_MS = 100000;

postRequest.setRetryPolicy(new DefaultRetryPolicy(
    MY_SOCKET_TIMEOUT_MS,
    DefaultRetryPolicy.DEFAULT_MAX_RETRIES,
    DefaultRetryPolicy.DEFAULT_BACKOFF_MULT));
requestQueue.add(postRequest);
}
}
}

```

#### 4.view\_doctor

```

package com.example.lenovo.aiskinexpert;

import android.content.SharedPreferences;
import android.preference.PreferenceManager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ListView;
import android.widget.Toast;

import com.android.volley.DefaultRetryPolicy;

```

```

import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONArray;
import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;

public class view_doctor extends AppCompatActivity {
    ListView l1;
    String[] did,nam,em,phn,qul;
    // ListView li;
    SharedPreferences sh;
    String ip, url, lid;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_view_doctor);
        l1=findViewById(R.id.list);
        l1 = (ListView) findViewById(R.id.list);
        sh = PreferenceManager.getDefaultSharedPreferences(getApplicationContext());
        ip = sh.getString("url", "");
        url = ip + "/and_view_doc";

        RequestQueue requestQueue = Volley.newRequestQueue(getApplicationContext());
        StringRequest postRequest = new StringRequest(Request.Method.POST, url,
            new Response.Listener<String>() {
                @Override
                public void onResponse(String response) {
                    // Toast.makeText(getApplicationContext(), response,
Toast.LENGTH_LONG).show();

                    try {
                        JSONObject jsonObj = new JSONObject(response);
                        if (jsonObj.getString("status").equalsIgnoreCase("ok")) {

                            JSONArray js = jsonObj.getJSONArray("data");//from python
                            nam = new String[js.length()];
                            em = new String[js.length()];
                            phn = new String[js.length()];
                            qul = new String[js.length()];
                            did = new String[js.length()];

```

```

        for (int i = 0; i < js.length(); i++) {
            JSONObject u = js.getJSONObject(i);
            nam[i] = u.getString("name");//dbcolumn name in double quotes
            em[i] = u.getString("email");
            phn[i] = u.getString("phone");
            qul[i] = u.getString("qualification");
            did[i] = u.getString("doctor_id");
        }
        ll.setAdapter(new costum_view_doctor(getApplicationContext(), nam,
em, phn, qul, did));//custom_view_service.xml and li is the listview object

    } else {
        Toast.makeText(getApplicationContext(), "Not found",
Toast.LENGTH_LONG).show();
    }

    } catch (Exception e) {
        Toast.makeText(getApplicationContext(), "Error" +
e.getMessage().toString(), Toast.LENGTH_SHORT).show();
    }
    },
    new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            // error
            Toast.makeText(getApplicationContext(), "eeeeee" + error.toString(),
Toast.LENGTH_SHORT).show();
        }
    }
    ) {

        //          value Passing android to python
        @Override
        protected Map<String, String> getParams() {
            SharedPreferences sh =
PreferenceManager.getDefaultSharedPreferences(getApplicationContext());
            Map<String, String> params = new HashMap<String, String>();
            params.put("id", sh.getString("lid","")); //passing to python
            return params;
        }
    };

    int MY_SOCKET_TIMEOUT_MS = 100000;

    postRequest.setRetryPolicy(new DefaultRetryPolicy(

```



```

        MY_SOCKET_TIMEOUT_MS,
        DefaultRetryPolicy.DEFAULT_MAX_RETRIES,
        DefaultRetryPolicy.DEFAULT_BACKOFF_MULT));
    requestQueue.add(postRequest);
}
}

```

## 5.Login.html

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Untitled Document</title>
</head>

<body>
<form id="form1" name="form1" method="post" action="">
  <table width="200" border="1">
    <tr>
      <th scope="row">Username</th>
      <td>
      </td>
    </tr>
    <tr>
      <th scope="row">Password</th>
      <td>
        <input type="password" name="textfield2" id="textfield2" />
      </td>
    </tr>
    <tr>
      <th colspan="2" scope="row"><input type="submit" name="button" id="button"
value="LOGIN" /></th>
    </tr>
    <tr>
      <th colspan="2" scope="row"><a href="/doctor_registration">Doctor
signup</a></th>
    </tr>
    <tr>
      <th colspan="2" scope="row"><a href="#">User Signup</a></th>
    </tr>
  </table>
</form>
</body>
</html>

```

## 6.Admin – index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <!-- SEO Meta Tags -->
  <meta name="description" content="Free mobile app HTML landing page template to help you build a great online presence for your app which will convert visitors into users">
  <meta name="author" content="Inovatik">

  <!-- OG Meta Tags to improve the way the post looks when you share the page on LinkedIn, Facebook, Google+ -->
  <meta property="og:site_name" content="" /> <!-- website name -->
  <meta property="og:site" content="" /> <!-- website link -->
  <meta property="og:title" content="" /> <!-- title shown in the actual shared post -->
  <meta property="og:description" content="" /> <!-- description shown in the actual shared post -->
  <meta property="og:image" content="" /> <!-- image link, make sure it's jpg -->
  <meta property="og:url" content="" /> <!-- where do you want your post to link to -->
  <meta property="og:type" content="article" />

  <!-- Website Title -->
  <title>AI SKIN</title>

  <!-- Styles -->
  <link href="https://fonts.googleapis.com/css?family=Montserrat:400,600,700" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css?family=Open+Sans:400,400i,700,700i" rel="stylesheet">
  <link href="/static/main_temp/css/bootstrap.css" rel="stylesheet">
  <link href="/static/main_temp/css/fontawesome-all.css" rel="stylesheet">
  <link href="/static/main_temp/css/swiper.css" rel="stylesheet">
  <link href="/static/main_temp/css/magnific-popup.css" rel="stylesheet">
  <link href="/static/main_temp/css/styles.css" rel="stylesheet">

  <!-- Favicon -->
  <link rel="icon" href="/static/main_temp/images/favicon.png">
</head>
<body data-spy="scroll" data-target=".fixed-top">

  <!-- Preloader -->
```

```

<div class="spinner-wrapper">
  <div class="spinner">
    <div class="bounce1"></div>
    <div class="bounce2"></div>
    <div class="bounce3"></div>
  </div>
</div>
<!-- end of preloader -->

<!-- Navbar -->
<nav class="navbar navbar-expand-md navbar-dark navbar-custom fixed-top">
  <!-- Text Logo - Use this if you don't have a graphic logo -->
  <a class="navbar-brand logo-text page-scroll" href="/admin_admin_home">AI
SKIN EXPERT</a>

  <!-- Image Logo -->
  {# <a class="navbar-brand logo-image" href="/"></a>#}

  <!-- Mobile Menu Toggle Button -->
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExampleDefault" aria-controls="navbarsExampleDefault" aria-
expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-awesome fas fa-bars"></span>
    <span class="navbar-toggler-awesome fas fa-times"></span>
  </button>
  <!-- end of mobile menu toggle button -->

  <div class="collapse navbar-collapse" id="navbarsExampleDefault">
    <ul class="navbar-nav ml-auto">
      <li class="nav-item">
        <a class="nav-link page-scroll" href="/admin_admin_home">HOME <span
class="sr-only">(current)</span></a>
      </li>

      <!-- Dropdown Menu -->
      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle page-scroll" href="#" id="drop1"
role="button" aria-haspopup="true" aria-expanded="false">DISEASE</a>
        <div class="dropdown-menu" aria-labelledby="drop1">
          <a class="dropdown-item" href="/admin_add_disease#preview"><span
class="item-text">Add Disease</span></a>
          <div class="dropdown-items-divide-hr"></div>
          <a class="dropdown-item" href="/admin_view_disease#preview"><span
class="item-text">View Disease</span></a>
        </div>
      </li>
      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle page-scroll" href="#" id="drop2"

```

```

role="button" aria-haspopup="true" aria-expanded="false">DOCTOR</a>
  <div class="dropdown-menu" aria-labelledby="drop2">
    <a class="dropdown-item"
href="/admin_approve_doctor#preview"><span class="item-text">Approve
doctor</span></a>
    <div class="dropdown-items-divide-hr"></div>
    <a class="dropdown-item"
href="/admin_view_approvedoctor#preview"><span class="item-text">View approved
doctor</span></a>
  </div>
</li>

```

```

<!-- end of dropdown menu -->
<li class="nav-item">
  <a class="nav-link page-scroll"
href="/admin_view_feedback#preview">FEEDBACKS</a>
</li>
<li class="nav-item">
  <a class="nav-link page-scroll"
href="/admin_view_user#preview">User</a>
</li>
<li class="nav-item">
  <a class="nav-link page-scroll" href="/logout">LOGOUT</a>
</li>

```

```

</ul>
</span>
</div>
</nav> <!-- end of navbar -->
<!-- end of navbar -->

```

```

<!-- Header -->
<header id="header" class="header">
  <div class="header-content">
    <div class="container">
      <div class="row">
        <div class="col-lg-6">
          <div class="text-container">
            <h1>AI SKIN EXPERT </h1><br><h6>AI Skin Expert app is a powerful
tool that empowers users to take control of their skin health.</h6>
            <p class="p-large">Skin Expert is one of the best solution for people who
have skin diseases</p>
            <a class="btn-solid-lg page-scroll" href="#your-link"><i class="fab fa-
apple"></i>APP STORE</a>
            <a class="btn-solid-lg page-scroll" href="#your-link"><i class="fab fa-
google-play"></i>PLAY STORE</a>
          </div>
        </div> <!-- end of col -->

```

```

        <div class="col-lg-6">
            <div class="image-container">
                
            </div> <!-- end of image-container -->
        </div> <!-- end of col -->
    </div> <!-- end of row -->
</div> <!-- end of container -->
</div> <!-- end of header-content -->
</header> <!-- end of header -->
<!-- end of header -->

```

```

<!-- Testimonials -->
<div class="slider-1">
    <div class="container">
        <div class="row">
            <div class="col-lg-12">

                <!-- Card Slider -->
                <div class="slider-container">
                    <div class="swiper-container card-slider">
                        <div class="swiper-wrapper">

                            <!-- Slide -->
                            <div class="swiper-slide">
                                <div class="card">
                                    
                                    <div class="card-body">
                                        <p class="testimonial-text">The app help to reduce healthcare
costs, as users can potentially avoid unnecessary doctor visits or medical procedures by
utilizing the app's diagnostic capabilities.</p>
                                        <p class="testimonial-author">Jude Thorn</p>
                                    </div>
                                </div>
                            </div>
                            <!-- end of swiper-slide -->
                            <!-- end of slide -->

                            <!-- Slide -->
                            <div class="swiper-slide">
                                <div class="card">
                                    
                                    <div class="card-body">
                                        <p class="testimonial-text">It can aid in the early detection and
diagnosis of skin conditions. The speed of this application is amazing!</p>
                                        <p class="testimonial-author">Roy Smith</p>
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>

```

```

</div> <!-- end of swiper-slide -->
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <div class="card">
    
    <div class="card-body">
      <p class="testimonial-text">This app has the potential of
becoming a mandatory tool in every marketer's day to day skin care.</p>
      <p class="testimonial-author">Elizabeth Olsen - Doctor</p>
    </div>
  </div>
</div> <!-- end of swiper-slide -->
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <div class="card">
    
    <div class="card-body">
      <p class="testimonial-text">It is important to note that an app
like AI Skin Expert should not replace the advice or treatment of a medical professional, and
users should always seek professional medical advice before starting any medication or
treatment.</p>
      <p class="testimonial-author">Tim Cook</p>
    </div>
  </div>
</div> <!-- end of swiper-slide -->
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <div class="card">
    
    <div class="card-body">
      <p class="testimonial-text">AI SKIN EXPERT's support team
is amazing. They've helped me with some issues and I am so grateful to them.</p>
      <p class="testimonial-author">Lindsay Spice</p>
    </div>
  </div>
</div> <!-- end of swiper-slide -->
<!-- end of slide -->

</div> <!-- end of swiper-wrapper -->

```

```

        <!-- Add Arrows -->
        <div class="swiper-button-next"></div>
        <div class="swiper-button-prev"></div>
        <!-- end of add arrows -->

    </div> <!-- end of swiper-container -->
</div> <!-- end of slider-container -->
<!-- end of card slider -->

</div> <!-- end of col -->
</div> <!-- end of row -->
</div> <!-- end of container -->
</div> <!-- end of slider-1 -->
<!-- end of testimonials -->

<!-- Video -->
<div id="preview" class="basic-1">
    <div class="container" style="background: #fff;color:#000;">
        <div class="row">
            <div class="col-lg-12 col-md-12">
                { % block body % }

                { % endblock % }
            </div>
        </div> <!-- end of row -->
    </div> <!-- end of container -->
</div> <!-- end of basic-1 -->
<!-- end of video -->

<!-- Screenshots -->
<div class="slider-2">
    <div class="container">
        <div class="row">
            <div class="col-lg-12">

                <!-- Image Slider -->
                <div class="slider-container">
                    <div class="swiper-container image-slider">
                        <div class="swiper-wrapper">

                            <!-- Slide -->
                            <!-- Slide -->
                            <div class="swiper-slide">
                                <a href="/static/main_temp/images/screenshot-1.png"
class="popup-link" data-effect="fadeIn">
                                    
    </a>
</div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
    <a href="/static/main_temp/images/screenshot-2.png"
class="popup-link" data-effect="fadeIn">
        
        </a>
    </div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
    <a href="/static/main_temp/images/screenshot-3.png"
class="popup-link" data-effect="fadeIn">
        
        </a>
    </div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
    <a href="/static/main_temp/images/screenshot-4.png"
class="popup-link" data-effect="fadeIn">
        
        </a>
    </div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
    <a href="/static/main_temp/images/screenshot-5.png"
class="popup-link" data-effect="fadeIn">
        
        </a>
    </div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
    <a href="/static/main_temp/images/screenshot-6.png"
class="popup-link" data-effect="fadeIn">
        <img class="img-fluid"

```



```

src="/static/main_temp/images/screenshot-6.png" alt="alternative">
    </a>
</div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
    <a href="/static/main_temp/images/screenshot-7.png"
class="popup-link" data-effect="fadeIn">
        
        </a>
    </div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
    <a href="/static/main_temp/images/screenshot-8.png"
class="popup-link" data-effect="fadeIn">
        
        </a>
    </div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
    <a href="/static/main_temp/images/screenshot-9.png"
class="popup-link" data-effect="fadeIn">
        
        </a>
    </div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
    <a href="/static/main_temp/images/screenshot-10.png"
class="popup-link" data-effect="fadeIn">
        
        </a>
    </div>
<!-- end of slide -->

</div> <!-- end of swiper-wrapper -->

<!-- Add Arrows -->
<div class="swiper-button-next"></div>
<div class="swiper-button-prev"></div>

```

```

        <!-- end of add arrows -->

        </div> <!-- end of swiper-container -->
    </div> <!-- end of slider-container -->
    <!-- end of image slider -->

    </div> <!-- end of col -->
    </div> <!-- end of row -->
    </div> <!-- end of container -->
</div> <!-- end of slider-2 -->
<!-- end of screenshots -->

<!-- Download -->
<div class="basic-4">
    <div class="container">
        <div class="row">
            <div class="col-lg-6 col-xl-5">
                <div class="text-container">
                    <h2>Download <span class="blue">AI SKIN EXPERT</span></h2>
                    <p class="p-large">The AI Skin Expert is an innovative mobile application
designed to help users identify skin conditions and diseases using artificial intelligence (AI)
technology.</p>
                    <a class="btn-solid-lg" href="#your-link"><i class="fab fa-apple"></i>APP STORE</a>
                    <a class="btn-solid-lg" href="#your-link"><i class="fab fa-google-play"></i>PLAY STORE</a>
                </div> <!-- end of text-container -->
            </div> <!-- end of col -->
            <div class="col-lg-6 col-xl-7">
                <div class="image-container">
                    
                </div> <!-- end of img-container -->
            </div> <!-- end of col -->
        </div> <!-- end of row -->
    </div> <!-- end of container -->
</div> <!-- end of basic-4 -->
<!-- end of download -->

<!-- Statistics -->
<div class="counter">
    <div class="container">
        <div class="row">
            <div class="col-lg-12">

                <!-- Counter -->
                <div id="counter">
                    <div class="cell">

```

```

        <div class="counter-value number-count" data-count="101">1</div>
        <p class="counter-info">Happy Users</p>
    </div>
    <div class="cell">
        <div class="counter-value number-count" data-count="70">1</div>
        <p class="counter-info">Issues Solved</p>
    </div>
    <div class="cell">
        <div class="counter-value number-count" data-count="59">1</div>
        <p class="counter-info">Good Reviews</p>
    </div>
    <div class="cell">
        <div class="counter-value number-count" data-count="127">1</div>
        <p class="counter-info">Cases</p>
    </div>
</div>
<!-- end of counter -->

</div> <!-- end of col -->
</div> <!-- end of row -->
</div> <!-- end of container -->
</div> <!-- end of counter -->
<!-- end of statistics -->

<!-- Contact -->
<div id="contact" class="form">
    <div class="container">
        <div class="row">
            <div class="col-lg-12">
                <h2>CONTACT</h2>
                <ul class="list-unstyled li-space-lg">
                    <li class="address">Don't hesitate to give us a call or just use the contact
form below</li>
                    <li><i class="fas fa-map-marker-alt"></i>22 Innovative, San Francisco,
CA 94043, US</li>
                    <li><i class="fas fa-phone"></i><a class="blue"
href="tel:003024630820">+81 720 2212</a></li>
                    <li><i class="fas fa-envelope"></i><a class="blue"
href="mailto:office@leno.com">office@expert.com</a></li>
                </ul>
            </div> <!-- end of col -->
        </div> <!-- end of row -->
        <div class="row">
            <div class="col-lg-6 offset-lg-3">

                <!-- Contact Form -->
                <form id="contactForm" data-toggle="validator" data-focus="false">
                    <div class="form-group">
                        <input type="text" class="form-control-input" id="cname" required>

```

```

        <label class="label-control" for="cname">Name</label>
        <div class="help-block with-errors"></div>
    </div>
    <div class="form-group">
        <input type="email" class="form-control-input" id="cemail"
required>
        <label class="label-control" for="cemail">Email</label>
        <div class="help-block with-errors"></div>
    </div>
    <div class="form-group">
        <textarea class="form-control-textarea" id="cmessage"
required></textarea>
        <label class="label-control" for="cmessage">Your message</label>
        <div class="help-block with-errors"></div>
    </div>
    <div class="form-group checkbox">
        <input type="checkbox" id="cterm" value="Agreed-to-Terms"
required>I have read and agree to Leno's stated conditions in <a href="privacy-
policy.html">Privacy Policy</a> and <a href="terms-conditions.html">Terms
Conditions</a>
        <div class="help-block with-errors"></div>
    </div>
    <div class="form-group">
        <button type="submit" class="form-control-submit-button">SUBMIT
MESSAGE</button>
    </div>
    <div class="form-message">
        <div id="cmgsSubmit" class="h3 text-center hidden"></div>
    </div>
</form>
<!-- end of contact form -->

</div> <!-- end of col -->
</div> <!-- end of row -->
</div> <!-- end of container -->
</div> <!-- end of form -->
<!-- end of contact -->

<!-- Footer -->
<div class="footer">
    <div class="container">
        <div class="row">
            <div class="col-md-4">
                <div class="footer-col">
                    <h4>About AI SKIN EXPERT</h4>
                    <p>The AI Skin Expert is an innovative mobile application designed to help
users identify skin conditions and diseases using artificial intelligence (AI) technology.</p>
                </div>
            </div> <!-- end of col -->

```

```

<div class="col-md-4">
  <div class="footer-col middle">
    <h4>Important Links</h4>
    <ul class="list-unstyled li-space-lg">
      <li class="media">
        <i class="fas fa-square"></i>
        <div class="media-body">Our business partners <a class="turquoise"
href="#your-link">startupguide.com</a></div>
      </li>
      <li class="media">
        <i class="fas fa-square"></i>
        <div class="media-body">Read our <a class="turquoise"
href="terms-conditions.html">Terms & Conditions</a>, <a class="turquoise"
href="privacy-policy.html">Privacy Policy</a></div>
      </li>
    </ul>
  </div>
</div> <!-- end of col -->
<div class="col-md-4">
  <div class="footer-col last">
    <h4>Social Media</h4>
    <span class="fa-stack">
      <a href="#your-link">
        <i class="fas fa-circle fa-stack-2x"></i>
        <i class="fab fa-facebook-f fa-stack-1x"></i>
      </a>
    </span>
    <span class="fa-stack">
      <a href="#your-link">
        <i class="fas fa-circle fa-stack-2x"></i>
        <i class="fab fa-twitter fa-stack-1x"></i>
      </a>
    </span>
    <span class="fa-stack">
      <a href="#your-link">
        <i class="fas fa-circle fa-stack-2x"></i>
        <i class="fab fa-google-plus-g fa-stack-1x"></i>
      </a>
    </span>
    <span class="fa-stack">
      <a href="#your-link">
        <i class="fas fa-circle fa-stack-2x"></i>
        <i class="fab fa-instagram fa-stack-1x"></i>
      </a>
    </span>
    <span class="fa-stack">
      <a href="#your-link">
        <i class="fas fa-circle fa-stack-2x"></i>
        <i class="fab fa-linkedin-in fa-stack-1x"></i>
      </a>
    </span>
  </div>
</div>

```

```

        </span>
    </div>
</div> <!-- end of col -->
</div> <!-- end of row -->
</div> <!-- end of container -->
</div> <!-- end of footer -->
<!-- end of footer -->

<!-- Copyright -->
<div class="copyright">
    <div class="container">
        <div class="row">
            <div class="col-lg-12">
                <p class="p-small">Copyright © AI SKIN EXPERT <a
href="http://www.inovatik.com">Inovatik</a></p>
            </div> <!-- end of col -->
        </div> <!-- enf of row -->
    </div> <!-- end of container -->
</div> <!-- end of copyright -->
<!-- end of copyright -->

<!-- Scripts -->
<script src="/static/main_temp/js/jquery.min.js"></script> <!-- jQuery for Bootstrap's
JavaScript plugins -->
<script src="/static/main_temp/js/popper.min.js"></script> <!-- Popper tooltip library
for Bootstrap -->
<script src="/static/main_temp/js/bootstrap.min.js"></script> <!-- Bootstrap
framework -->
<script src="/static/main_temp/js/jquery.easing.min.js"></script> <!-- jQuery Easing
for smooth scrolling between anchors -->
<script src="/static/main_temp/js/swiper.min.js"></script> <!-- Swiper for image and
text sliders -->
<script src="/static/main_temp/js/jquery.magnific-popup.js"></script> <!-- Magnific
Popup for lightboxes -->
<script src="/static/main_temp/js/morphext.min.js"></script> <!-- Morphetext rotating
text in the header -->
<script src="/static/main_temp/js/validator.min.js"></script> <!-- Validator.js -
Bootstrap plugin that validates forms -->
<script src="/static/main_temp/js/scripts.js"></script> <!-- Custom scripts -->
</body>
</html>

```

## 7.approve doctor

```
{% extends "admin/index.html" %}
{% block body %}
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Untitled Document</title>
</head>

<body>
<form id="form1" name="form1" method="post" action="">
  <table class="table table-bordered" border="1">
    <tr>
      <th scope="col">#</th>
      <th scope="col">name</th>
      <th scope="col">email</th>
      <th scope="col">phone_no</th>
      <th scope="col">qualification</th>
      <th scope="col">&nbsp;</th>
    </tr>
    {% for i in data %}
    <tr>
      <td>{{ loop.index }}</td>
      <td>&nbsp;{{ i.name }}</td>
      <td>&nbsp;{{ i.email }}</td>
      <td>&nbsp;{{ i.phone }}</td>
      <td>&nbsp;{{ i.qualification }}</td>
      <td><p><a href="/admin_approve_btn/{{ i.login_id }}" class="btn btn-
primary">approve</a></p>
      <p><a href="/admin_reject_btn/{{ i.login_id }}" class="btn btn-
danger">reject</a></p></td>
    </tr>
    {% endfor %}
  </table>
</form>
</body>
</html>
{% endblock %}
```

## 8.doctor\_index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```

<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-
fit=no">

<!-- SEO Meta Tags -->
<meta name="description" content="Free mobile app HTML landing page template
to help you build a great online presence for your app which will convert visitors into
users">
<meta name="author" content="Inovatik">

<!-- OG Meta Tags to improve the way the post looks when you share the page on
LinkedIn, Facebook, Google+ -->
<meta property="og:site_name" content="" /> <!-- website name -->
<meta property="og:site" content="" /> <!-- website link -->
<meta property="og:title" content="" /> <!-- title shown in the actual shared post -->
<meta property="og:description" content="" /> <!-- description shown in the actual
shared post -->
<meta property="og:image" content="" /> <!-- image link, make sure it's jpg -->
<meta property="og:url" content="" /> <!-- where do you want your post to link to -->
<meta property="og:type" content="article" />

<!-- Website Title -->
<title>AI SKIN</title>

<!-- Styles -->
<link href="https://fonts.googleapis.com/css?family=Montserrat:400,600,700"
rel="stylesheet">
<link href="https://fonts.googleapis.com/css?family=Open+Sans:400,400i,700,700i"
rel="stylesheet">
<link href="/static/main_temp/css/bootstrap.css" rel="stylesheet">
<link href="/static/main_temp/css/fontawesome-all.css" rel="stylesheet">
<link href="/static/main_temp/css/swiper.css" rel="stylesheet">
<link href="/static/main_temp/css/magnific-popup.css" rel="stylesheet">
<link href="/static/main_temp/css/styles.css" rel="stylesheet">

<!-- Favicon -->
<link rel="icon" href="/static/main_temp/images/favicon.png">
</head>
<body data-spy="scroll" data-target=".fixed-top">

<!-- Preloader -->
<div class="spinner-wrapper">
  <div class="spinner">
    <div class="bounce1"></div>
    <div class="bounce2"></div>
    <div class="bounce3"></div>
  </div>
</div>
<!-- end of preloader -->

```



```

<!-- Navbar -->
<nav class="navbar navbar-expand-md navbar-dark navbar-custom fixed-top">
  <!-- Text Logo - Use this if you don't have a graphic logo -->
  <a class="navbar-brand logo-text page-scroll" href="/admin_admin_home">AI
SKIN EXPERT</a>

  <!-- Image Logo -->
  {#    <a class="navbar-brand logo-image" href="/"></a>#}

  <!-- Mobile Menu Toggle Button -->
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarsExampleDefault" aria-controls="navbarsExampleDefault" aria-
expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-awesome fas fa-bars"></span>
    <span class="navbar-toggler-awesome fas fa-times"></span>
  </button>
  <!-- end of mobile menu toggle button -->

  <div class="collapse navbar-collapse" id="navbarsExampleDefault">
    <ul class="navbar-nav ml-auto">
      <li class="nav-item">
        <a class="nav-link page-scroll" href="/doctor_doctor_home">HOME
<span class="sr-only">(current)</span></a>
      </li>
      <li class="nav-item">
        <a class="nav-link page-scroll"
href="/doctor_view_profile#preview">PROFILE</a>
      </li>

      <!-- Dropdown Menu -->
      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle page-scroll" href="#" id="drop1"
role="button" aria-haspopup="true" aria-expanded="false">SCHEDULE</a>
        <div class="dropdown-menu" aria-labelledby="drop1">
          <a class="dropdown-item"
href="/doctor_add_schedule#preview"><span class="item-text">Add
Schedule</span></a>
          <div class="dropdown-items-divide-hr"></div>
          <a class="dropdown-item"
href="/doctor_view_schedule#preview"><span class="item-text">View
Schedule</span></a>
        </div>
      </li>

      <li class="nav-item dropdown">
        <a class="nav-link dropdown-toggle page-scroll" href="#" id="drop1"
role="button" aria-haspopup="true" aria-expanded="false">APPOINTMENTS</a>
        <div class="dropdown-menu" aria-labelledby="drop1">

```

```

        <a class="dropdown-item"
href="/doctor_view_appointments#preview"><span class="item-text">Todays
Bookings</span></a>
        <div class="dropdown-items-divide-hr"></div>
        <a class="dropdown-item"
href="/doctor_view_schedule#preview"><span class="item-text">View
Bookings</span></a>
        <div class="dropdown-items-divide-hr"></div>
        <a class="dropdown-item"
href="/doctor_view_previous_patients#preview"><span class="item-text">Previous
Bookings</span></a>
    </div>
</li>
<li class="nav-item">
    <a class="nav-link page-scroll"
href="/doctor_view_disease_doctor#preview">DISEASE </a>
</li>
<li class="nav-item">
    <a class="nav-link page-scroll"
href="/doctor_view_reviews#preview">REVIEW</a>
</li>

```

```

<li class="nav-item dropdown">
    <a class="nav-link dropdown-toggle page-scroll" href="#" id="drop1"
role="button" aria-haspopup="true" aria-expanded="false">PREDICTION</a>
    <div class="dropdown-menu" aria-labelledby="drop1">
        <a class="dropdown-item"
href="/doctor_prediction_image#preview"><span class="item-
text">IMAGE</span></a>
        <div class="dropdown-items-divide-hr"></div>
        <a class="dropdown-item" href="/add_symptoms#preview"><span
class="item-text">SYMPTOMS</span></a>
    </div>
</li>

```

```

{#
    <li class="nav-item">#}
{#
    <a class="nav-link page-scroll"
href="/doctor_prediction_image#preview">PREDICTION IMAGE</a>#}
{#
    </li>#}
    <li class="nav-item">
        <a class="nav-link page-scroll"
href="/doctor_change_password#preview"> CHANGE PASSWORD</a>
    </li>
    <li class="nav-item">
        <a class="nav-link page-scroll" href="/logout">LOGOUT</a>
    </li>

```

```

    </ul>
  </span>
</div>
</nav> <!-- end of navbar -->
<!-- end of navbar -->

<!-- Header -->
<header id="header" class="header">
  <div class="header-content">
    <div class="container">
      <div class="row">
        <div class="col-lg-6">
          <div class="text-container">
            <h1>AI SKIN EXPERT <br>FOR <span id="js-rotating">Doctors,
Patients</span></h1>
            <p class="p-large">Skin Expert is one of the best solution for people who
have skin diseases</p>
            <a class="btn-solid-lg page-scroll" href="#your-link"><i class="fab fa-
apple"></i>APP STORE</a>
            <a class="btn-solid-lg page-scroll" href="#your-link"><i class="fab fa-
google-play"></i>PLAY STORE</a>
          </div>
        </div> <!-- end of col -->
        <div class="col-lg-6">
          <div class="image-container">
            
          </div> <!-- end of image-container -->
        </div> <!-- end of col -->
      </div> <!-- end of row -->
    </div> <!-- end of container -->
  </div> <!-- end of header-content -->
</header> <!-- end of header -->
<!-- end of header -->

<!-- Testimonials -->
<div class="slider-1">
  <div class="container">
    <div class="row">
      <div class="col-lg-12">

        <!-- Card Slider -->
        <div class="slider-container">
          <div class="swiper-container card-slider">
            <div class="swiper-wrapper">

              <!-- Slide -->
              <div class="swiper-slide">

```

```

<div class="card">
  
  <div class="card-body">
    <p class="testimonial-text">The app help to reduce healthcare
costs, as users can potentially avoid unnecessary doctor visits or medical procedures by
utilizing the app's diagnostic capabilities.</p>
    <p class="testimonial-author">Jude Thorn </p>
  </div>
</div>
</div> <!-- end of swiper-slide -->
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <div class="card">
    
    <div class="card-body">
      <p class="testimonial-text">It can aid in the early detection and
diagnosis of skin conditions.The speed of this application is amazing!</p>
      <p class="testimonial-author">Roy Smith </p>
    </div>
  </div>
</div> <!-- end of swiper-slide -->
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <div class="card">
    
    <div class="card-body">
      <p class="testimonial-text">This app has the potential of
becoming a mandatory tool in every individuals's day to day skin care.</p>
      <p class="testimonial-author">Elizabeth Olsen -Doctor </p>
    </div>
  </div>
</div> <!-- end of swiper-slide -->
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <div class="card">
    
    <div class="card-body">
      <p class="testimonial-text">It is important to note that an app
like AI Skin Expert should not replace the advice or treatment of a medical professional, and
users should always seek professional medical advice before starting any medication or

```

```

treatment.</p>
        <p class="testimonial-author">Tim Cook </p>
    </div>
</div>
</div> <!-- end of swiper-slide -->
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
    <div class="card">
        
        <div class="card-body">
            <p class="testimonial-text">AI SKIN EXPERT's support team
is amazing. They've helped me with some issues and I am so grateful to them.</p>
            <p class="testimonial-author">Lindsay Spice</p>
        </div>
    </div>
</div> <!-- end of swiper-slide -->
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
    <div class="card">
        
        <div class="card-body">
            <p class="testimonial-text">Doctor is really good and one thing
which differentiate her is her simplicity and sincerely listening to patents problem and overall
cost including appointment and medicine comparatively economical.</p>
            <p class="testimonial-author">Ann Black</p>
        </div>
    </div>
</div> <!-- end of swiper-slide -->
<!-- end of slide -->

</div> <!-- end of swiper-wrapper -->

<!-- Add Arrows -->
<div class="swiper-button-next"></div>
<div class="swiper-button-prev"></div>
<!-- end of add arrows -->

</div> <!-- end of swiper-container -->
</div> <!-- end of slider-container -->
<!-- end of card slider -->

</div> <!-- end of col -->
</div> <!-- end of row -->
</div> <!-- end of container -->

```

```

</div> <!-- end of slider-1 -->
<!-- end of testimonials -->

```

```

<!-- Video -->
<div id="preview" class="basic-1">
  <div class="container" style="background: #fff;color:#000;">
    <div class="row">
      <div class="col-lg-12 col-md-12">
        { % block body % }

        { % endblock % }
      </div>
    </div> <!-- end of row -->
  </div> <!-- end of container -->
</div> <!-- end of basic-1 -->
<!-- end of video -->

```

```

<!-- Screenshots -->
<div class="slider-2">
  <div class="container">
    <div class="row">
      <div class="col-lg-12">

        <!-- Image Slider -->
        <div class="slider-container">
          <div class="swiper-container image-slider">
            <div class="swiper-wrapper">

              <!-- Slide -->
              <div class="swiper-slide">
                <a href="/static/main_temp/images/screenshot-1.png"
class="popup-link" data-effect="fadeIn">
                  
                </a>
              </div>
              <!-- end of slide -->

              <!-- Slide -->
              <div class="swiper-slide">
                <a href="/static/main_temp/images/screenshot-2.png"
class="popup-link" data-effect="fadeIn">
                  
                </a>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>

```

```

<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <a href="/static/main_temp/images/screenshot-3.png"
class="popup-link" data-effect="fadeIn">
    
  </a>
</div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <a href="/static/main_temp/images/screenshot-4.png"
class="popup-link" data-effect="fadeIn">
    
  </a>
</div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <a href="/static/main_temp/images/screenshot-5.png"
class="popup-link" data-effect="fadeIn">
    
  </a>
</div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <a href="/static/main_temp/images/screenshot-6.png"
class="popup-link" data-effect="fadeIn">
    
  </a>
</div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <a href="/static/main_temp/images/screenshot-7.png"
class="popup-link" data-effect="fadeIn">
    
  </a>
</div>

```

```

<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <a href="/static/main_temp/images/screenshot-8.png"
class="popup-link" data-effect="fadeIn">
    
  </a>
</div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <a href="/static/main_temp/images/screenshot-9.png"
class="popup-link" data-effect="fadeIn">
    
  </a>
</div>
<!-- end of slide -->

<!-- Slide -->
<div class="swiper-slide">
  <a href="/static/main_temp/images/screenshot-10.png"
class="popup-link" data-effect="fadeIn">
    
  </a>
</div>
<!-- end of slide -->

</div> <!-- end of swiper-wrapper -->

<!-- Add Arrows -->
<div class="swiper-button-next"></div>
<div class="swiper-button-prev"></div>
<!-- end of add arrows -->

</div> <!-- end of swiper-container -->
</div> <!-- end of slider-container -->
<!-- end of image slider -->

</div> <!-- end of col -->
</div> <!-- end of row -->
</div> <!-- end of container -->
</div> <!-- end of slider-2 -->
<!-- end of screenshots -->

```



```

<!-- Download -->
<div class="basic-4">
  <div class="container">
    <div class="row">
      <div class="col-lg-6 col-xl-5">
        <div class="text-container">
          <h2>Download <span class="blue">AI SKIN EXPERT</span></h2>
          <p class="p-large">The AI Skin Expert is an innovative mobile application
designed to help users identify skin conditions and diseases using artificial intelligence (AI)
technology.</p>
          <a class="btn-solid-lg" href="#your-link"><i class="fab fa-apple"></i>APP STORE</a>
          <a class="btn-solid-lg" href="#your-link"><i class="fab fa-google-play"></i>PLAY STORE</a>
        </div> <!-- end of text-container -->
      </div> <!-- end of col -->
    <div class="col-lg-6 col-xl-7">
      <div class="image-container">
        
      </div> <!-- end of img-container -->
    </div> <!-- end of col -->
  </div> <!-- end of row -->
</div> <!-- end of container -->
</div> <!-- end of basic-4 -->
<!-- end of download -->

```

```

<!-- Statistics -->
<div class="counter">
  <div class="container">
    <div class="row">
      <div class="col-lg-12">

        <!-- Counter -->
        <div id="counter">
          <div class="cell">
            <div class="counter-value number-count" data-count="101">1</div>
            <p class="counter-info">Happy Users</p>
          </div>
          <div class="cell">
            <div class="counter-value number-count" data-count="70">1</div>
            <p class="counter-info">Issues Solved</p>
          </div>
          <div class="cell">
            <div class="counter-value number-count" data-count="100">1</div>
            <p class="counter-info">Good Reviews</p>
          </div>
          <div class="cell">
            <div class="counter-value number-count" data-count="150">1</div>

```

```

        <p class="counter-info">Cases</p>
    </div>
</div>
<!-- end of counter -->

</div> <!-- end of col -->
</div> <!-- end of row -->
</div> <!-- end of container -->
</div> <!-- end of counter -->
<!-- end of statistics -->

<!-- Contact -->
<div id="contact" class="form">
    <div class="container">
        <div class="row">
            <div class="col-lg-12">
                <h2>CONTACT</h2>
                <ul class="list-unstyled li-space-lg">
                    <li class="address">Don't hesitate to give us a call or just use the contact
form below</li>
                    <li><i class="fas fa-map-marker-alt"></i>22 Innovative, San Francisco,
CA 94043, US</li>
                    <li><i class="fas fa-phone"></i><a class="blue"
href="tel:003024630820">+81 720 2212</a></li>
                    <li><i class="fas fa-envelope"></i><a class="blue"
href="mailto:office@leno.com">office@expert.com</a></li>
                </ul>
            </div> <!-- end of col -->
        </div> <!-- end of row -->
        <div class="row">
            <div class="col-lg-6 offset-lg-3">

                <!-- Contact Form -->
                <form id="contactForm" data-toggle="validator" data-focus="false">
                    <div class="form-group">
                        <input type="text" class="form-control-input" id="cname" required>
                        <label class="label-control" for="cname">Name</label>
                        <div class="help-block with-errors"></div>
                    </div>
                    <div class="form-group">
                        <input type="email" class="form-control-input" id="cemail"
required>
                        <label class="label-control" for="cemail">Email</label>
                        <div class="help-block with-errors"></div>
                    </div>
                    <div class="form-group">
                        <textarea class="form-control-textarea" id="cmessage"
required></textarea>
                        <label class="label-control" for="cmessage">Your message</label>

```

```

        <div class="help-block with-errors"></div>
    </div>
    <div class="form-group checkbox">
        <input type="checkbox" id="cterm" value="Agreed-to-Terms"
required>I have read and agree to AI SKIN EXPERT's stated conditions in <a
href="privacy-policy.html">Privacy Policy</a> and <a href="terms-
conditions.html">Terms Conditions</a>
        <div class="help-block with-errors"></div>
    </div>
    <div class="form-group">
        <button type="submit" class="form-control-submit-button">SUBMIT
MESSAGE</button>
    </div>
    <div class="form-message">
        <div id="cmmsgSubmit" class="h3 text-center hidden"></div>
    </div>
</form>
<!-- end of contact form -->

</div> <!-- end of col -->
</div> <!-- end of row -->
</div> <!-- end of container -->
</div> <!-- end of form -->
<!-- end of contact -->

<!-- Footer -->
<div class="footer">
    <div class="container">
        <div class="row">
            <div class="col-md-4">
                <div class="footer-col">
                    <h4>About AI SKIN EXPERT</h4>
                    <p>The AI Skin Expert is an innovative mobile application designed to help
users identify skin conditions and diseases using artificial intelligence (AI) technology.</p>
                </div>
            </div> <!-- end of col -->
            <div class="col-md-4">
                <div class="footer-col middle">
                    <h4>Important Links</h4>
                    <ul class="list-unstyled li-space-lg">
                        <li class="media">
                            <i class="fas fa-square"></i>
                            <div class="media-body">Our business partners <a class="turquoise"
href="#your-link">startupguide.com</a></div>
                        </li>
                        <li class="media">
                            <i class="fas fa-square"></i>
                            <div class="media-body">Read our <a class="turquoise"
href="terms-conditions.html">Terms & Conditions</a>, <a class="turquoise"

```

```

href="privacy-policy.html">Privacy Policy</a></div>
    </li>
</ul>
</div>
</div> <!-- end of col -->
<div class="col-md-4">
    <div class="footer-col last">
        <h4>Social Media</h4>
        <span class="fa-stack">
            <a href="#your-link">
                <i class="fas fa-circle fa-stack-2x"></i>
                <i class="fab fa-facebook-f fa-stack-1x"></i>
            </a>
        </span>
        <span class="fa-stack">
            <a href="#your-link">
                <i class="fas fa-circle fa-stack-2x"></i>
                <i class="fab fa-twitter fa-stack-1x"></i>
            </a>
        </span>
        <span class="fa-stack">
            <a href="#your-link">
                <i class="fas fa-circle fa-stack-2x"></i>
                <i class="fab fa-google-plus-g fa-stack-1x"></i>
            </a>
        </span>
        <span class="fa-stack">
            <a href="#your-link">
                <i class="fas fa-circle fa-stack-2x"></i>
                <i class="fab fa-instagram fa-stack-1x"></i>
            </a>
        </span>
        <span class="fa-stack">
            <a href="#your-link">
                <i class="fas fa-circle fa-stack-2x"></i>
                <i class="fab fa-linkedin-in fa-stack-1x"></i>
            </a>
        </span>
    </div>
</div> <!-- end of col -->
</div> <!-- end of row -->
</div> <!-- end of container -->
</div> <!-- end of footer -->
<!-- end of footer -->

<!-- Copyright -->
<div class="copyright">
    <div class="container">
        <div class="row">

```

```

        <div class="col-lg-12">
            <p class="p-small">Copyright © AI SKIN EXPERT <a
href="http://www.inovatik.com">Inovatik</a></p>
        </div> <!-- end of col -->
    </div> <!-- enf of row -->
</div> <!-- end of container -->
</div> <!-- end of copyright -->
<!-- end of copyright -->

<!-- Scripts -->
<script src="/static/main_temp/js/jquery.min.js"></script> <!-- jQuery for Bootstrap's
JavaScript plugins -->
<script src="/static/main_temp/js/popper.min.js"></script> <!-- Popper tooltip library
for Bootstrap -->
<script src="/static/main_temp/js/bootstrap.min.js"></script> <!-- Bootstrap
framework -->
<script src="/static/main_temp/js/jquery.easing.min.js"></script> <!-- jQuery Easing
for smooth scrolling between anchors -->
<script src="/static/main_temp/js/swiper.min.js"></script> <!-- Swiper for image and
text sliders -->
<script src="/static/main_temp/js/jquery.magnific-popup.js"></script> <!-- Magnific
Popup for lightboxes -->
<script src="/static/main_temp/js/morphext.min.js"></script> <!-- Morphtext rotating
text in the header -->
<script src="/static/main_temp/js/validator.min.js"></script> <!-- Validator.js -
Bootstrap plugin that validates forms -->
<script src="/static/main_temp/js/scripts.js"></script> <!-- Custom scripts -->
</body>
</html>

```

## 9.Doctor- view\_appointment

```

{ % extends "doctor/doctor_index.html" % }
{ % block body % }
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Untitled Document</title>
</head>

<body>
<form id="form1" name="form1" method="post" action="">
    <table class="table table-bordered" border="1">
        <tr>
            <th scope="col">#</th>

```

```

<th scope="col">Token No</th>
<th scope="col">Name</th>
<th scope="col">Phone</th>
<th scope="col">Age</th>
<th scope="col">Gender</th>
<th scope="col">&nbsp;</th>
</tr>
{% for i in data %}
<tr>
<td>&nbsp;{{ loop.index }}</td>
<td>&nbsp;{{ i.token_no }}</td>
<td>{{ i.name }}</td>
<td>&nbsp;{{ i.phone }}</td>
<td>{{ i.age }}&nbsp;</td>
<td>&nbsp;{{ i.gender }}</td>
<td><a href="#" class="btn btn-primary">Chat</a></td>
</tr>
{% endfor %}
</table>
</form>
</body>
</html>
{% endblock %}

```

## **6. SYSTEM TESTING**

## 6.1 TESTING AND EVALUATION

Testing is a process of executing a program with the intent of finding an error. Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design and coding. Testing includes verifications of the basic logic of each program and verification that the entire system works properly. Testing demonstrates that software functions appear to be working according to specification. In addition, data collected as testing is conducted provides a good indication of software quality as a whole. The debugging process is the most unpredictable part of testing process. Testing begins at the module level and works towards the integration of the entire computer-based system. Testing and debugging are different activities, but any testing includes debugging strategy for software testing must accommodate low level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system function, against customer requirements. No testing is complete without verification and validation part. The goals of verification and validation activities are to access and improve the quality of work products generated during the development and modification of the software. There are two types of verification: life cycle verification and formal verification. Life cycle verification is the process of determining the degree to which the products of the given phase of the development cycle fulfil the specification established during the prior process. Formal verification is the rigorous mathematical demonstration that source code conforms to its specifications. Validation is a process of evaluating software at the end of the software development process to determine conformance with the requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The primary objectives, when we test software are the following:

- Testing is a process of exceeding with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.
- A successful test is one uncovers undiscovered errors.

Thus, testing plays a very critical role in determining the reliability and efficiency of the software and hence is very important stage in software development. Tests are to be conducted on the software to evaluate its performance under a number of conditions. Ideally, it should so at the level



of each module and also when all of them are integrated to form the completed system. Software testing is done at different levels.

## 6.2 TESTING STRATEGIES

A strategy for software testing integrates software test case design method into a well-planned series of steps that result in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote a customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project progresses.

**The general characteristics of software testing strategies are:**

- Testing begins at the component level and works “outward” toward the integration of the entire computer system.
- Different testing techniques are appropriate at different points in time.

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

A strategy must provide guidance for the practitioner and set of milestones for the manager. Because the step on the test strategy occurs at a time when deadline pressure begins to rise, progress must be measurable and problem must surface as early as possible.

The software team’s approach to testing is defining a plan that describes an overall strategy and a procedure that defines specific testing steps and tests that will be conducted. In the proposed system, if the administrator makes any attempt to login to the application without entering his password, then the system will not allow the user to login to the application.

## **6.3 TESTING TECHNIQUES**

The various testing techniques are given below:

### **6.3.1 WHITE BOX TESTING**

White-box testing is also called as glass-box testing, is a test case design method that goes to the control structure of the procedural design to derive

test cases. Using white box testing methods, the software engineer can derive test cases that,

- ✓ Guarantee that all independent paths within a module have been exercised at Least once.
- ✓ Exercise all logical decision on their true and false sides.
- ✓ Execute all loops at their boundaries and within their operational sides.
- ✓ Exercise internal data structure to ensure their validity.

White box testing was successfully conducted on our system. All independent paths within a module have been executed at least once and all logical decisions have been exercised on their true and false sides.

### **6.3.2 BLACK BOX TESTING**

Black-box testing is also called as behavioural testing, focuses on the functional requirement of the software. It is a complimentary approach that is likely uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories.

- ✓ Incorrect or missing functions.
- ✓ Interface errors.
- ✓ Error on data structures or external database access.
- ✓ Behaviour or performance errors.
- ✓ Initialization and termination errors.

Black box testing was successfully conducted on your system. The system was divided into a number of modules and testing was conducted on each module. We have tested the system for incorrect or missing functions, interface and performance errors.

### **6.3.3 UNIT TESTING**

Unit testing comprises the set of tests performed by an individual programmer prior to the integration of the system. Testing removes residual bugs and improves the reliability of the system.

Testing allows the developer to find out the design faults if any, and enable correction if needed. Exhaustive unit testing has to be carried out to ensure the validity of the data. In order to successfully test the entire package, unit testing is carried out. Each module was tested as when it was developed. Thus it proved easier to conduct minute testing operation and correct them then and there.

### **6.3.4 INTEGRATION TESTING**

Bottom-up integration is the traditional strategy used to integrate the component of a software system into a functional whole. Bottom-up integration consists of unit testing, followed by subsystem testing and followed by testing of the entire system. Unit testing has the goal of discovering errors in the individual parts of the system.

Parts are tested in isolation from one another in an artificial environment known as “Test Harness”, which consists of driver programs and data necessary to exercise the modules. Unit testing should be as exhaustive as possible to ensure that each representative case handled by each module has been tested. Unit testing is eased by a system structure that is composed of small loosely coupled modules.

Both control and data interfaces must be tested. Large software system may require several levels of subsystem testing. Lower level subsystems are successively combined to form higher level subsystems. In most software systems, exhaustive testing of subsystem capabilities is not feasible due to the combination complexity of the module interfaces. Therefore, test cases must be carefully chosen to exercise the interfaces in the desired manner.

### **6.3.5 ACCEPTANCE TESTING**

Acceptance testing involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements. It

is not unusual for two sets of acceptance test to be run, those developed by the quality group and those developed by the customer.

In addition to the functional and performance tests, stress tests are performed to determine the limitation of the system. For example, a compiler might be tested to determine the effect of the symbol table overflow, or real-time system might be tested to determine the effect of simultaneous arrival of numerous high priority interrupts.

#### **6.3.6 OUTPUT TESTING**

Output testing of the proposed system is important since no system could be useful if it does not produce the required output.

The output format on the screen is found to be correct, as the format was designed in the system design phase according to the user needs. For the hard copy also the output comes out as the specified requirements by the user. Hence output testing doesn't result in any correction on the system.

## **7. SYSTEM IMPLEMENTATION AND DEPLOYMENT**

Implementation is the process of deploying the new system in the operational environment. Proper implementation is essential to provide a reliable system to meet the organizational requirement. There are four types of implementation methods. They are Direct Changeover, Phased Implementation, Parallel Run and Pilot Approach. The most commonly used implementation methods are Pilot Approach and Parallel Run.

The system which is developed as a web application hence the other functions as normal application, as usual some web development technologies are used in the implementation of the project. The language I selected to program this software is PHP. The reason I selected PHP is that is a simple and powerful language that especially developed to create web application.

Technologies used in the development of the software are:

- ✓ Programming language – Python
- ✓ DBMS – MySQL server
- ✓ Development tool – Py charm
- ✓ Development platform – Windows 10

The front end is HTML, and CSS and back end is MySQL Server and Python. The system developed on Pycharm in Windows 10 operating system.

## **8. CONCLUSION**

The “Ai Skin Expert” has been developed for all given conditions and it is found working effectively under the all the circumstances that may arise in the real environment. The software has been developed to reducing the operator work.

This system is user friendly and is well efficient to make easy interaction with the users of system. The system is done with an insight into the necessary modifications that may be required in the future. Hence the system can be maintained successfully without much work.

### **8.1 FUTURE ENHANCEMENT**

As a future venture, it is suggested to make some changes to provide more services and to provide information at right time in right manner.

The current system is designed to detect helmet and mask violations in the college premises. It can only detect the violations taking place within the college.

In the future the system can be extended to detect violations in a larger area like the village or Panchayat or even more. More features like detecting whether the students have worn ID cards or not can also be added to the system. All the functions have been done carefully and successfully in the software, and if any development is necessary in future, it can be done without affecting the design by adding additional modules to the system.

## **9. REFERENCE**

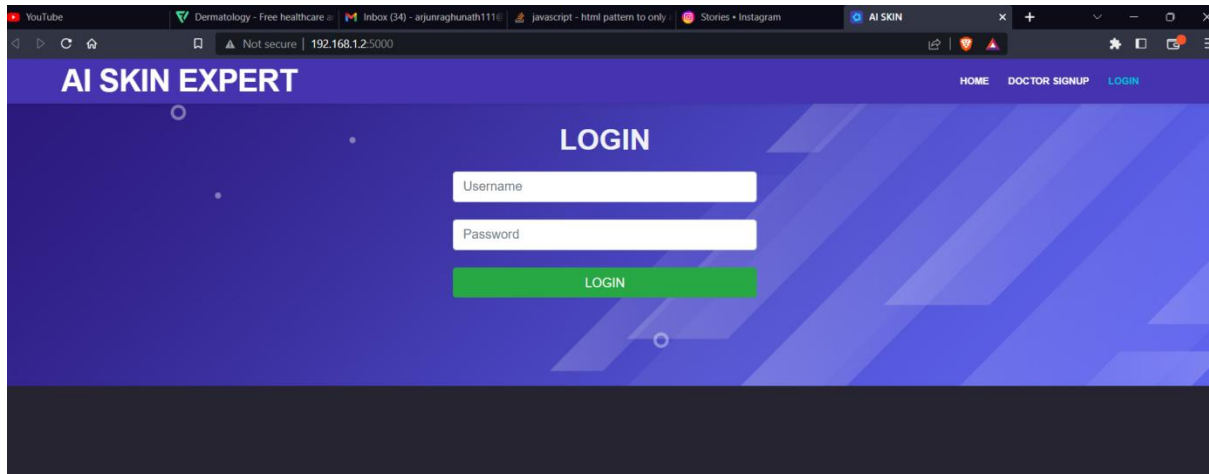
- Google.com
- Youtube

## **10. APPENDIX**



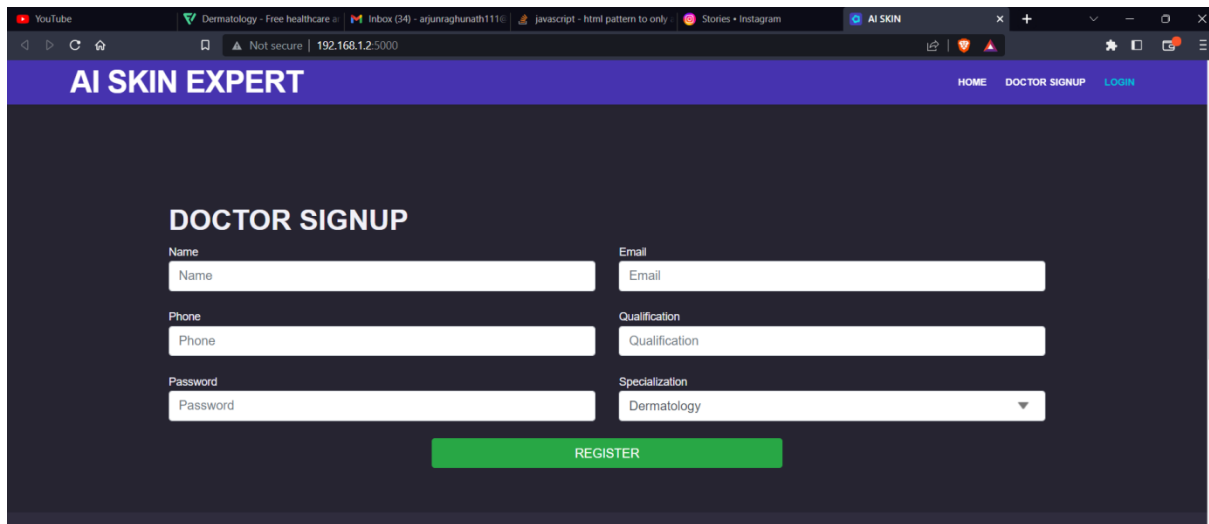
## 10.1 Screenshots

### 10.1.1 Homepage:



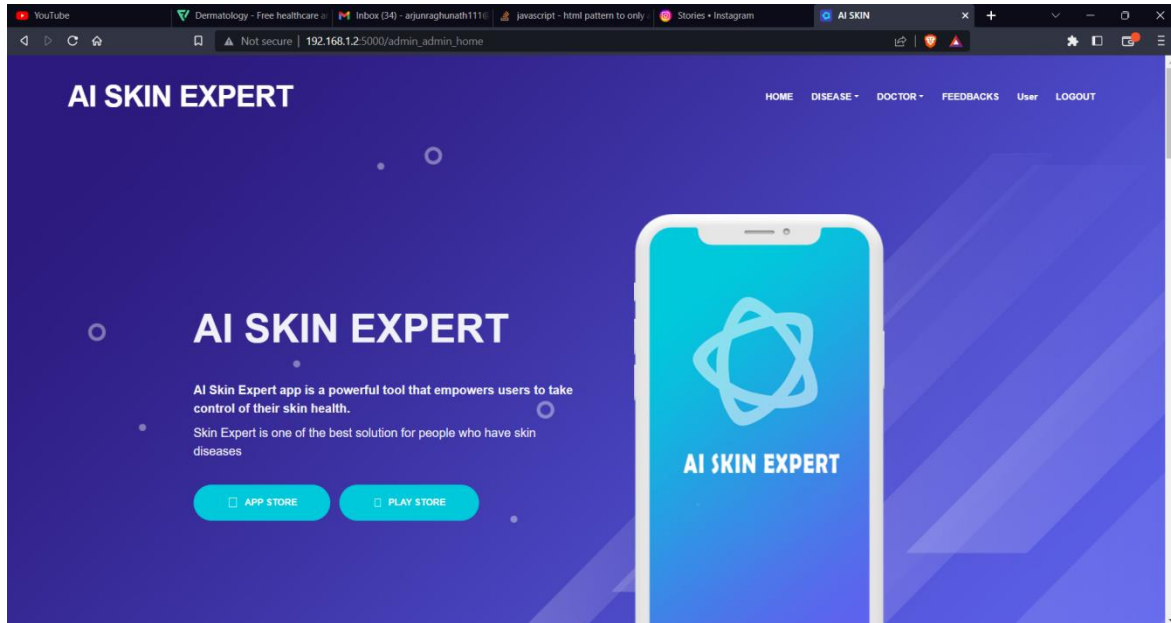
The screenshot shows the homepage of the 'AI SKIN EXPERT' application. The header is a dark blue bar with the text 'AI SKIN EXPERT' on the left and navigation links 'HOME', 'DOCTOR SIGNUP', and 'LOGIN' on the right. The main content area has a dark blue background with a subtle geometric pattern. In the center, there is a white 'LOGIN' form with two input fields: 'Username' and 'Password', and a green 'LOGIN' button below them.

### 10.1.2 Doctor SignUp:

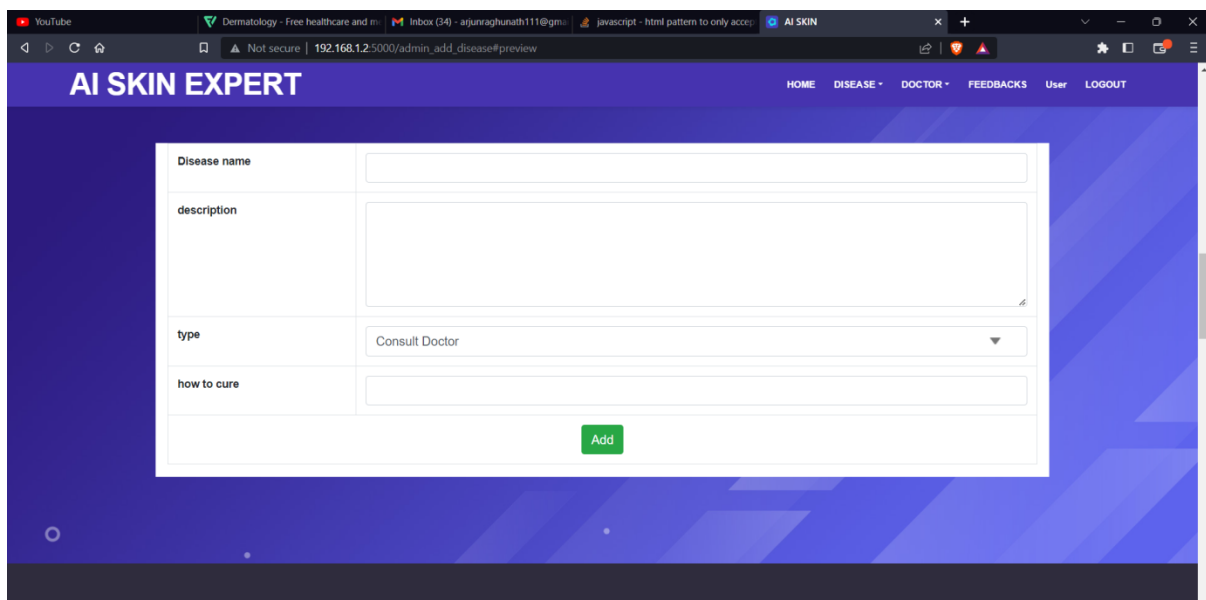


The screenshot shows the 'DOCTOR SIGNUP' page of the 'AI SKIN EXPERT' application. The header is identical to the homepage. The main content area has a dark blue background. The 'DOCTOR SIGNUP' form is centered and consists of several input fields arranged in two columns: 'Name', 'Email', 'Phone', 'Qualification', 'Password', and 'Specialization'. The 'Specialization' field is a dropdown menu with 'Dermatology' selected. A green 'REGISTER' button is positioned at the bottom center of the form.

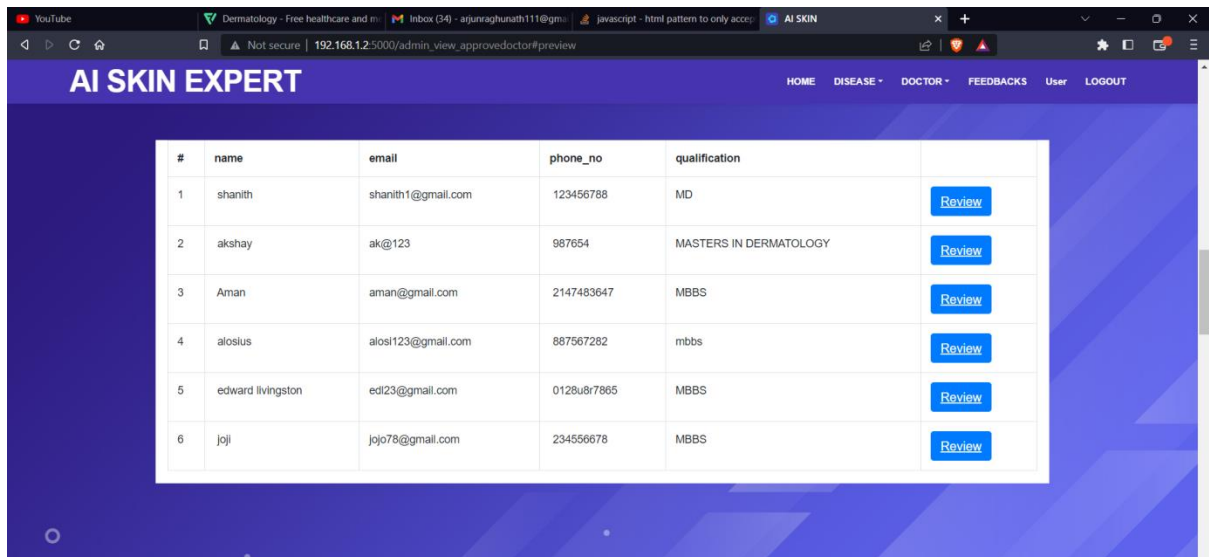
### 10.1.3 :Admin Homepage



### 10.1.4 Admin- Add disease:

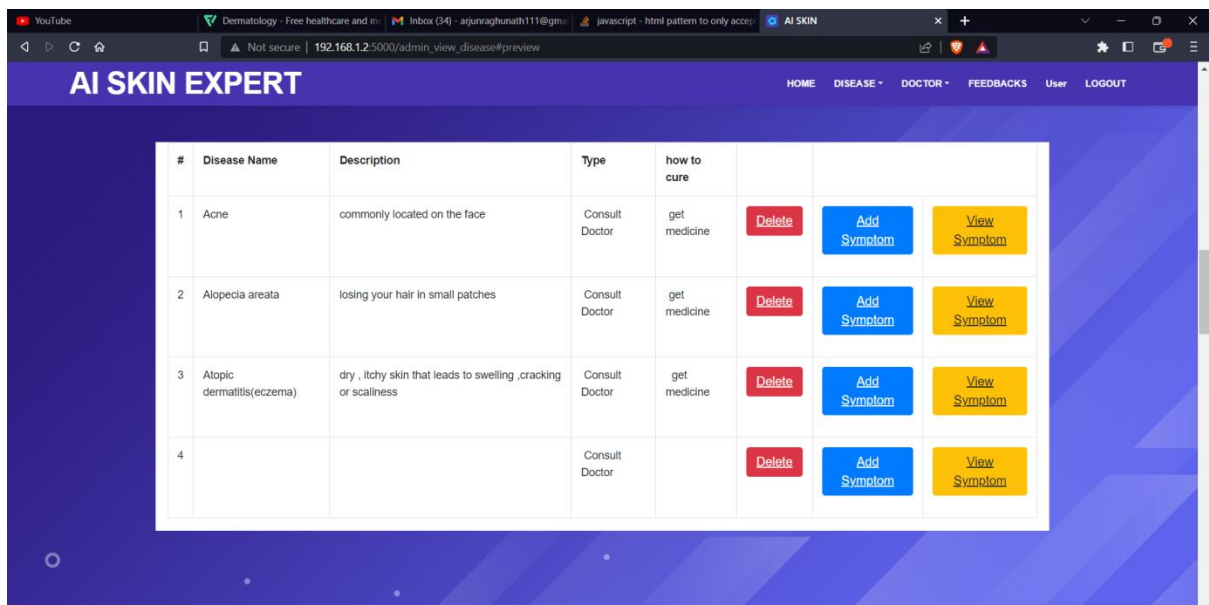


### 10.1.5 Admin- View doctors:



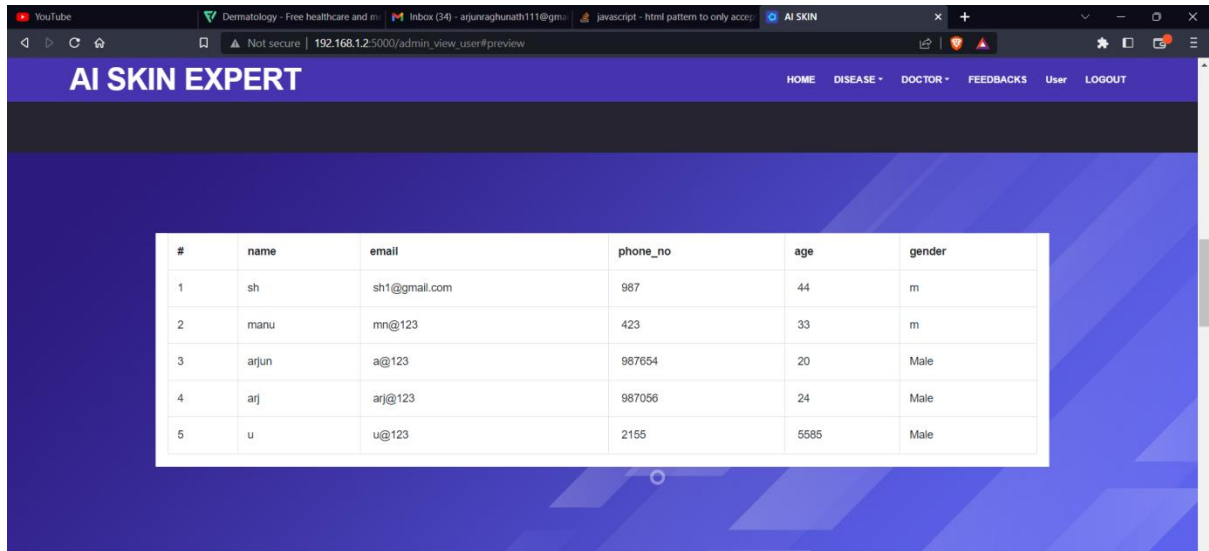
#	name	email	phone_no	qualification	
1	shanith	shanith1@gmail.com	123456788	MD	<a href="#">Review</a>
2	akshay	ak@123	987654	MASTERS IN DERMATOLOGY	<a href="#">Review</a>
3	Aman	aman@gmail.com	2147483647	MBBS	<a href="#">Review</a>
4	alosius	alos123@gmail.com	887567282	mbbs	<a href="#">Review</a>
5	edward livingston	ed123@gmail.com	0128u8r7865	MBBS	<a href="#">Review</a>
6	joji	jojo78@gmail.com	234556678	MBBS	<a href="#">Review</a>

### 10.1.6 Admin –view diseases:



#	Disease Name	Description	Type	how to cure			
1	Acne	commonly located on the face	Consult Doctor	get medicine	<a href="#">Delete</a>	<a href="#">Add Symptom</a>	<a href="#">View Symptom</a>
2	Alopecia areata	losing your hair in small patches	Consult Doctor	get medicine	<a href="#">Delete</a>	<a href="#">Add Symptom</a>	<a href="#">View Symptom</a>
3	Atopic dermatitis(eczema)	dry, itchy skin that leads to swelling ,cracking or scaliness	Consult Doctor	get medicine	<a href="#">Delete</a>	<a href="#">Add Symptom</a>	<a href="#">View Symptom</a>
4			Consult Doctor		<a href="#">Delete</a>	<a href="#">Add Symptom</a>	<a href="#">View Symptom</a>

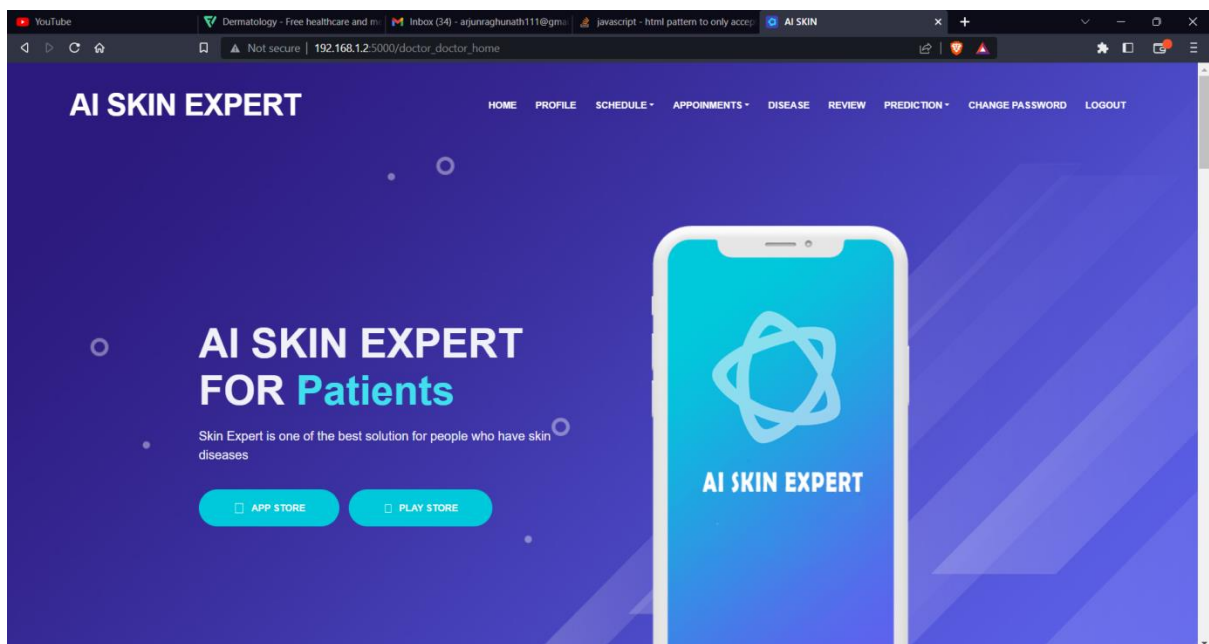
### 10.1.7 Admin- view users:



The screenshot shows the 'AI SKIN EXPERT' Admin interface. The header includes navigation links: HOME, DISEASE, DOCTOR, FEEDBACKS, User, and LOGOUT. The main content area displays a table with the following data:

#	name	email	phone_no	age	gender
1	sh	sh1@gmail.com	987	44	m
2	manu	mn@123	423	33	m
3	arjun	a@123	987654	20	Male
4	arj	arj@123	987056	24	Male
5	u	u@123	2155	5585	Male

### 10.1.8 Doctor- Homepage:



The screenshot shows the 'AI SKIN EXPERT' Doctor Homepage. The header includes navigation links: HOME, PROFILE, SCHEDULE, APPOINTMENTS, DISEASE, REVIEW, PREDICTION, CHANGE PASSWORD, and LOGOUT. The main content area features a large blue background with a smartphone displaying the app logo and text:

**AI SKIN EXPERT FOR Patients**

Skin Expert is one of the best solution for people who have skin diseases

[APP STORE](#) [PLAY STORE](#)

### 10.1.9 Doctor- view profile:

The screenshot shows a web browser window with the URL `192.168.1.2:5000/doctor_view_profile#preview`. The page title is "AI SKIN EXPERT". The navigation menu includes: HOME, PROFILE, SCHEDULE, APPOINTMENTS, DISEASE, REVIEW, PREDICTION, CHANGE PASSWORD, and LOGOUT. The main content area displays a form for viewing a doctor's profile:

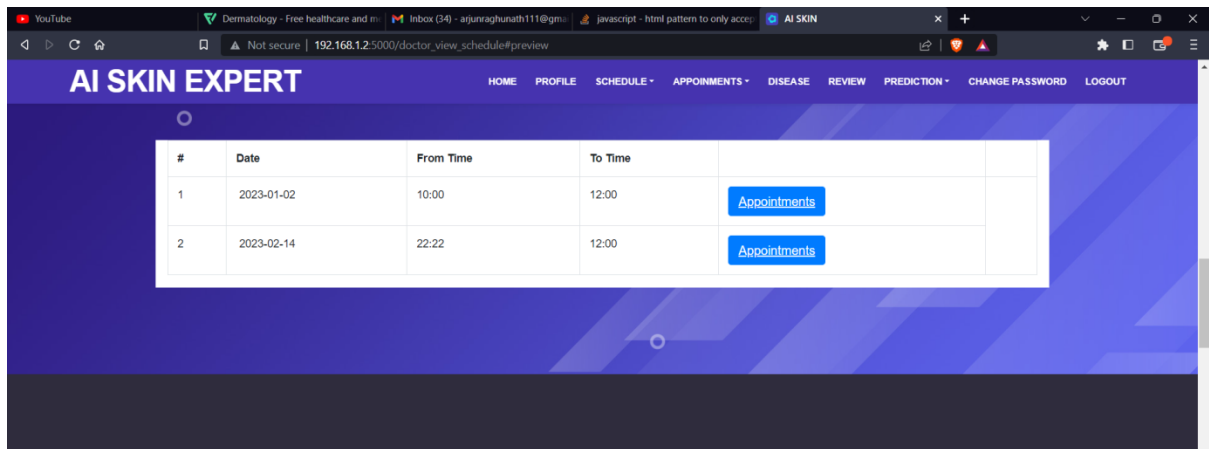
Name	shanith
Email	shanith1@gmail.com
Phone	123456788
Qualification	MD
<input type="button" value="update"/>	

### 10.1.10 Doctor- Add schedule:

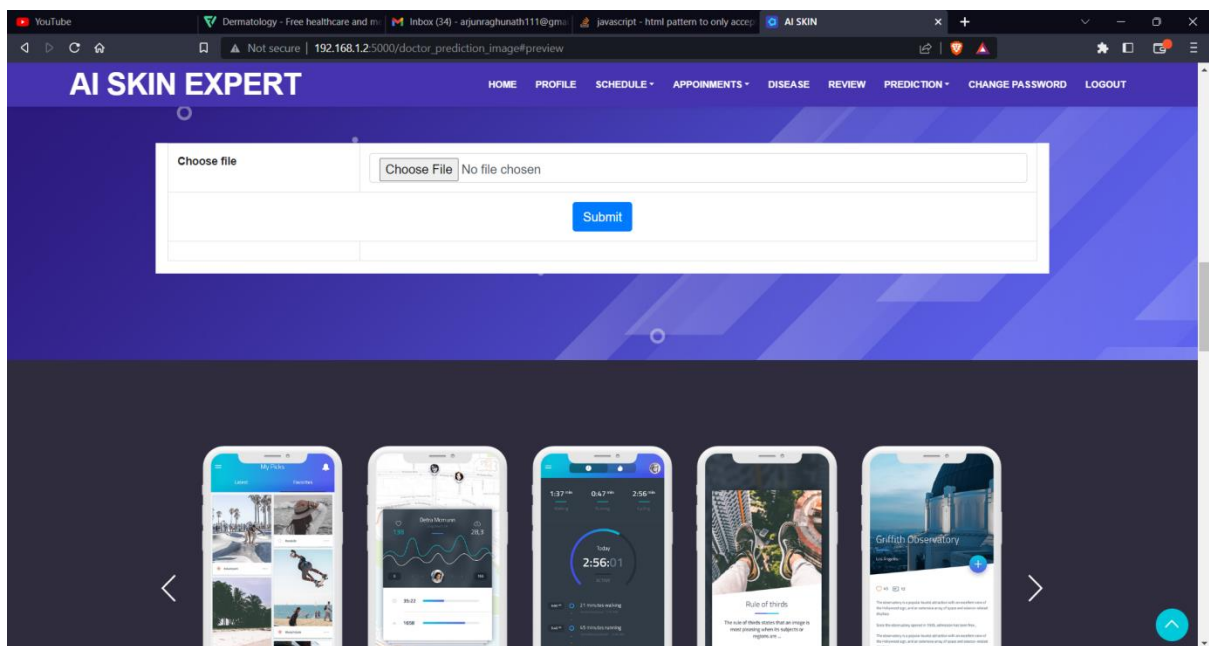
The screenshot shows a web browser window with the URL `192.168.1.2:5000/doctor_add_schedule#preview`. The page title is "AI SKIN EXPERT". The navigation menu includes: HOME, PROFILE, SCHEDULE, APPOINTMENTS, DISEASE, REVIEW, PREDICTION, CHANGE PASSWORD, and LOGOUT. The main content area displays a form for adding a schedule:

Date	mm/dd/yyyy	<input type="button" value="calendar icon"/>
From time	--:--	<input type="button" value="clock icon"/>
To time	--:--	<input type="button" value="clock icon"/>
<input type="button" value="ADD"/>		

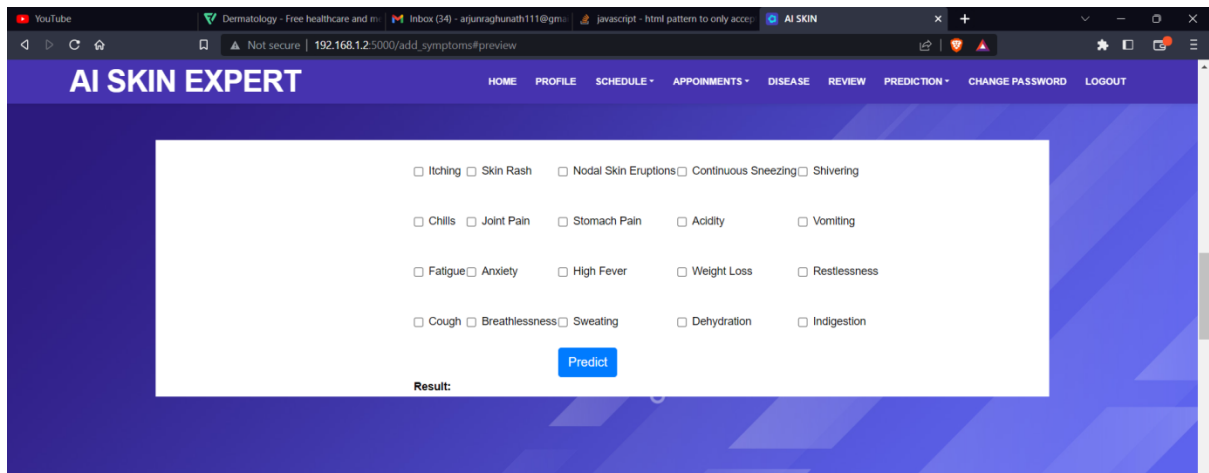
### 10.1.11 Doctor- Homepage:



### 10.1.12 Doctor- prediction by image:

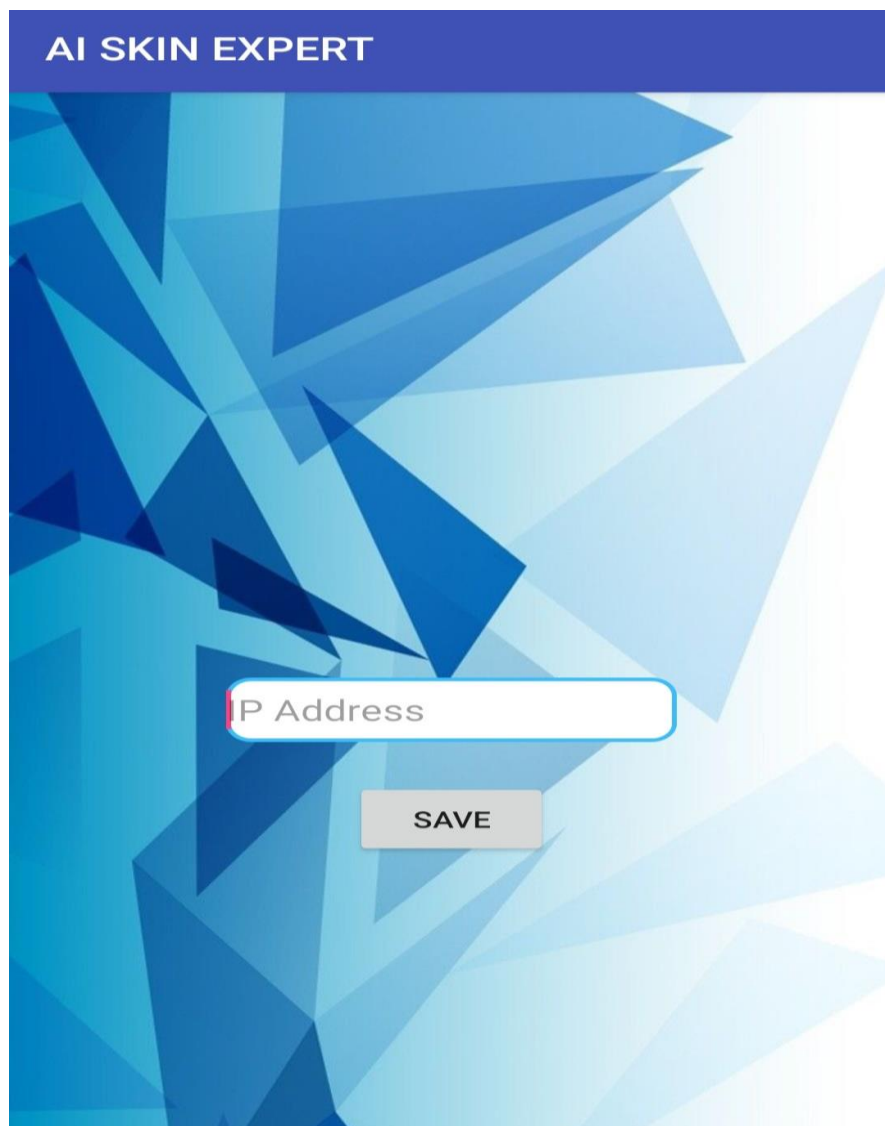


### 10.1.13 Doctor- Predict by symptoms:

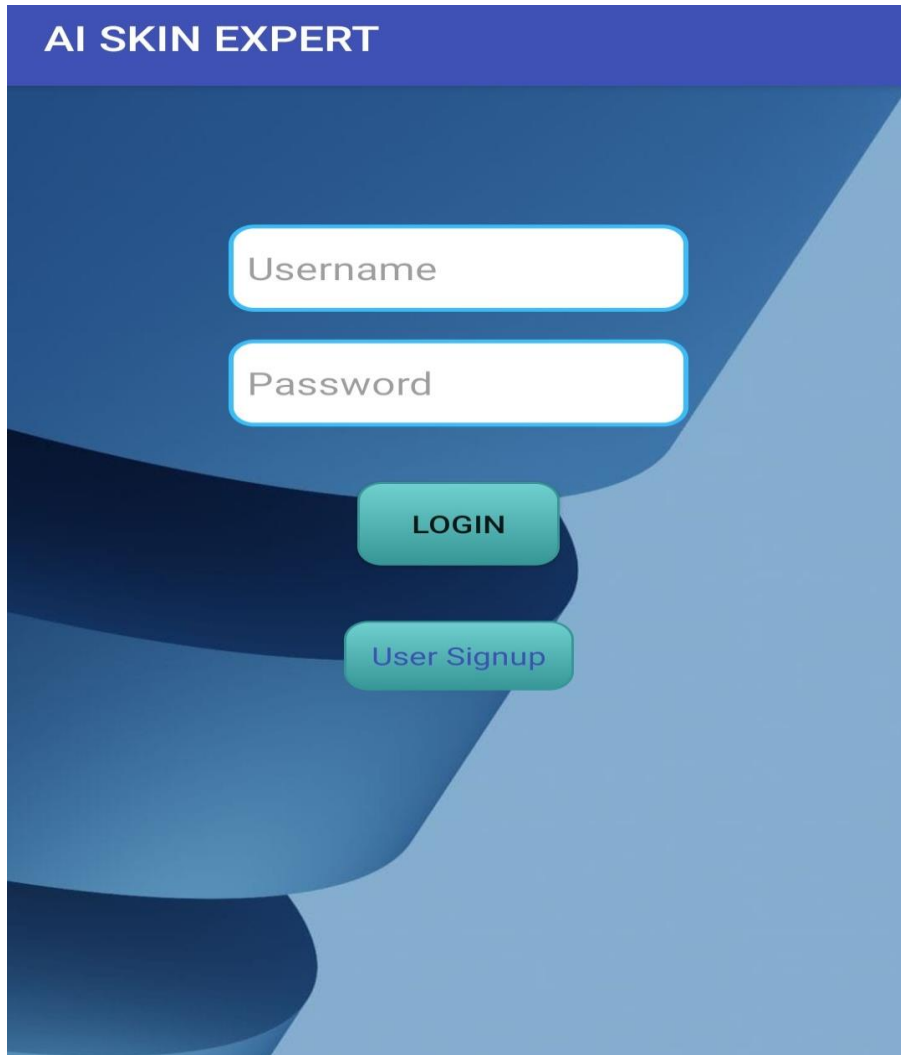


## ANDROID

User IP connection



## Login page



The image shows a login page for 'AI SKIN EXPERT'. The page has a dark blue header with the text 'AI SKIN EXPERT' in white. Below the header, there are two white input fields with blue borders, labeled 'Username' and 'Password'. Below these fields are two green buttons with white text: 'LOGIN' and 'User Signup'. The background is a dark blue gradient with a stylized, abstract shape on the left side.

AI SKIN EXPERT

Username

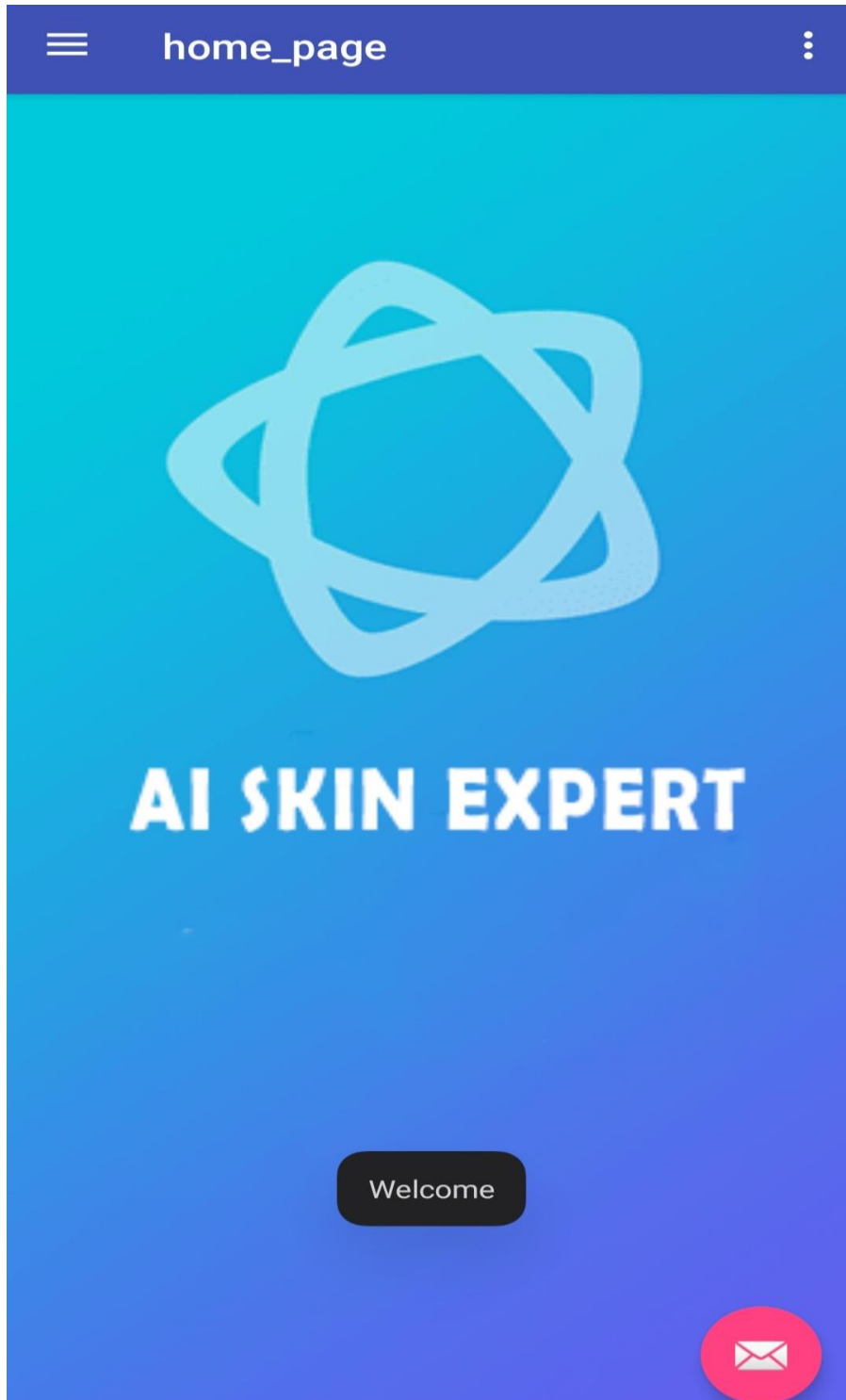
Password

LOGIN

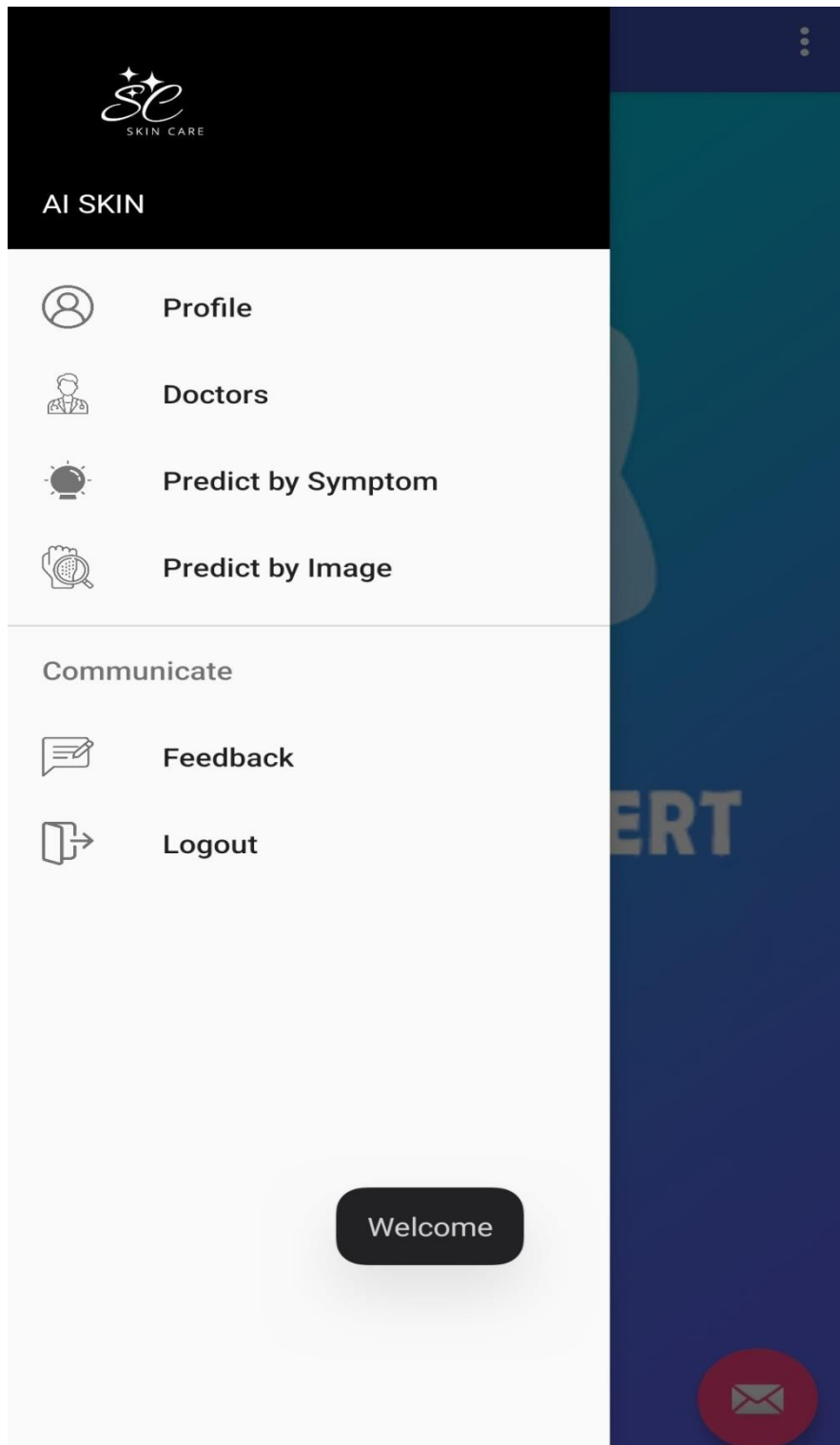
User Signup



User home page:



Home page functions :



User view profile:

**AI SKIN EXPERT**

<b>Name</b>	arj
<b>Email</b>	arj@1234
<b>Phone</b>	987056
<b>Age</b>	24
<b>Gender</b>	Male

view doctor

**AI SKIN EXPERT**

Name	shanith
Email	shanith1@gmail.com
Phone	123456788
Qualification	MD
<div>REVIEWVIEW SLOTCHAT</div>	

Name	main
Email	krjgn@gsmil.com
Phone	2345
Qualification	MBBS
<div>REVIEWVIEW SLOTCHAT</div>	

Name	akshay
Email	ak@123
Phone	987654
Qualification	MASTERS IN DERMATOLOGY
<div>REVIEWVIEW SLOTCHAT</div>	

## Prediction by symptom

**AI SKIN EXPERT**


Choose Symptoms

- ☐ Itching
- ☐ Skin Rash
- ☐ Continuous Sneezing
- ☐ Nodal Skin Eruptions
- ☐ Shivering
- ☐ Chills
- ☐ Joint Pain
- ☐ Stomach Pain
- ☐ Skin Rash
- ☐ Acidity
- ☐ Vomiting
- ☐ Fatigue
- ☐ Anxiety
- ☐ High Fever
- ☐ Weight Loss
- ☐ Restlessness
- ☐ Cough
- ☐ Breathlessness
- ☐ Sweating
- ☐ Dehydration
- ☐ Indigestion

PREDICT

Prediction by image :

**AI SKIN EXPERT**

A blue icon representing an image upload. It features a light blue rounded rectangle containing a darker blue silhouette of a landscape with a mountain and a sun. A blue arrow points upwards from the top right corner of the rectangle.

PREDICT

Result

Feedback :

**AI SKIN EXPERT**

Feedback

SEND

**A PROJECT REPORT ON**  
**BFIT**  
**AI BASED FITNESS CENTER**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ADARSH JOE**

**REG.NO: DB20BCAR17**

**ALBIN M B**

**REG.NO: DB20BCAR20**

**ANANDHU KRISHNA**

**REG.NO: DB20BCAR21**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**

**A PROJECT REPORT ON**  
**BFIT**  
**AI BASED FITNESS CENTER**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ADARSH JOE**

**REG.NO: DB20BCAR17**

**ALBIN M B**

**REG.NO: DB20BCAR20**

**ANANDHU KRISHNA**

**REG.NO: DB20BCAR21**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**

**DON BOSCO ARTS & SCIENCE  
COLLEGEANGADIKADAVUIRITTY, KANNUR**



**CERTIFICATE**

Certified that this report titled **BFIT-AI BAESD FITNESS CENTER** is a bonafide record of the project work done by **Adarsh Joe (Reg.No: DB20BCAR17), Albin M B (Reg.No:DB20BCAR20) and Anandhu Krishna (Reg.No:DB20BCAR21)** under the supervision and guidance, towards partial fulfilment of the requirement for award of the degree of bachelor of computer application (BCA) of the Kannur university.

**Project Guide**

.....

Angadikadavu

Date:

**Head of the Department**

.....

External Examiner

1.

2.



## **Declaration**

We **ADARSH JOE, ALBIN M B and ANANDHU KRISHNA** sixth semester BCA student of Don Bosco Arts & Science College, Angadikadavu, under Kannur University do hereby declare that the project entitled “**BFIT-AI BASED FITNESS CENTER**” is the original work carried out by me in the sixth semester under the supervision of **Ms. Sruthi Nimesh**, Lecture of the Department of BCA, Don Bosco Arts & Science College, Angadikadavu, in partial fulfilment of the requirement for the award of the degree Bachelor of Computer Application, Kannur University.

Angadikadavu

Date

ADARSH JOE

ALBIN M B

ANANDHU KRISHNA

## **ACKNOWLEDGEMENT**

First of all we thank the lord almighty for his immense grace and blessings showered on us at every stages of this work. I am greatly indebted to our Principal Fr. Dr. Francis Karackat SDB, Don Bosco Arts & Science College, Angadikadavu for providing the opportunity to take up this project as part of my curriculum.

We deeply indebted to our project guide Ms. Sruthi Nimesh, lectures of department of BCA, for her assistance and valuable suggestions as guide. She made this project a reality.

We express our sincere thanks to Mrs. Sindu PM, Mr. Hebin Layola, Mrs. Fincy Cyriac and Mrs.Vineetha Mathew, lecturers of department of BCA, for their valuable suggestions during the course of this project. Their critical suggestions helped me to improve the project work.

Acknowledging the efforts of everyone, their chivalrous help in the course of the project preparation and their willingness to collaborate with the work, their magnanimity through lucid technical details lead to the successful completion of my project.

We would like to express our sincere thanks to all our friends, colleagues, parents and all those who have directly or indirectly assisted during this work.

# CONTENTS

<b>chapters</b>	<b>contents</b>	<b>Page No</b>
1	Introduction	1
2	System Analysis	7
2.1	Existing System	8
2.1.1	Disadvantage Of Existing System	9
2.2	Proposed System	9
2.2.1	Advantage Of Proposed System	10
2.3	Feasibility Study	10
2.3.1	Economic Feasibility	11
2.3.2	Technical Feasibility	11
2.3.3	Behavioural Feasibility	11
2.4	System Specification	12
2.4.1	Software Specification	12
2.4.2	Hardware Specification	13
2.5	Identification Of Actors	13
2.6	Identification Of Use Cases	14
2.6.1	Use Cases For The Actor Admin	14
2.6.2	Use Cases For The Actor Physician	15
2.6.3	Use Cases For The Actor Gym Instructor	15
2.6.4	Use Cases For The Actor User	16
2.6.5	Use Cases For The Actor Doctor	16
2.6.6	Use Case Diagram	17

3	SYSTEM DESIGN	22
3.1	Introduction	23
3.2	Database Design	23
3.3	Table Design	24
3.4	AI Based Fitness Center	25
3.5	Data Flow Diagram	32
3.6	ER Diagram	42
4	CODING	44
4.1	Input Interface	45
4.2	Output Interface	45
4.3	Software Description	45
4.3.1	HTML	46
4.3.3	CSS	46
4.3.3	JavaScript	47
4.3.4	MySQL	48
4.3.5	Python	51
4.3.6	Flask	52
5	CODING PAGES	53
5.1	Admin Page	54
6	System Testing	62
6.1	Testing And Evaluation	63
6.2	Testing Strategies	64
6.3	Testing Techniques	65
6.3.1	White Box Testing	65

6.3.2	Black Box Testing	65
6.3.3	Unit Testing	66
6.3.4	Integration Testing	66
6.3.5	Acceptance Testing	67
6.3.6	Output Testing	67
7	SYSTEM IMPLEMENTATION AND DEPLOYMENT	68
8	CONCLUSION	69
9	REFERANCE	70
10	APPENDIX	71

# **CHAPTER I**

## **INTRODUCTION**

## **1.1 Project Overview**

Gyms have become an essential part of our lives, providing the best exercise and body-building facilities to our society. Gym Management Systems typically offer a wealth of solutions for every day operation, streamlining processes in a way that can enhance the customer experience. From online gym scheduling and automated billing to administrative tasks, the software pulls all data into one place so that you can run your business more efficiently. The purpose of the project is to build an application program to reduce the manual work for managing the Fitness Center, Fitness Master, Health, Employee, It tracks all the details about the Employee, Member, Diets. We also incorporates a heart disease prediction functionality for the user within the system using Random forest classifier. Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction.

## **NEED FOR THE SYSTEM**

Gym management software is generally designed to streamline operations so that all of these tasks can be in one place. In a world where technological advances occur so quickly, it can feel like a challenge to keep up. Maintaining the records on paper is very difficult so, it is necessary to have a computerized system that manages all these issues. Gym Management Systems typically offer a wealth of solutions for every day operation, streamlining processes in a way that can enhance the customer experience. From online gym scheduling and automated billing to administrative tasks, the software pulls all data into one place so that you can run your business more efficiently. The purpose of the project is to build an application program to reduce the manual work for managing the Fitness Center, Fitness Master, Health, Employee, It tracks all the details about the Employee, Member and Diets.

## **OBJECTIVES AND SCOPE**

In a world where technological advances occur so quickly, it can feel like a challenge to keep up. But gym and fitness clubs can maximize business potential through gym management software. This type of software has gone beyond just processing membership payments and additional admin tools. It can help you manage all facets

of the business, retain and engage members, and, most importantly – grow. Gym management software can also be referred to as club management software, fitness software, or gym scheduling software. Regardless of the nomenclature, these platforms all share similar feature sets and are used for the same purposes. Gym management software helps fitness owners and operators manage their class and trainer scheduling, keep track of their members, communicate with clients, and process payments. One of the most important benefits of a fitness software is that they can help increase member retention. Assuming clients enjoy the workouts and the sense of community at your studio, if the software is user friendly when they're scheduling from a mobile or on their computer, then they'll continue to come back.

The main objectives behind the development of this project are:

- 1) To manage the details of Fitness Master, Employee, Member.
- 2) To manage all the information about Fitness Center, Health, Diets.
- 3) To predict heart disease

The scope of this project is high as it reduces manual work and utilises the resources in an efficient manner with minimum usage of time.

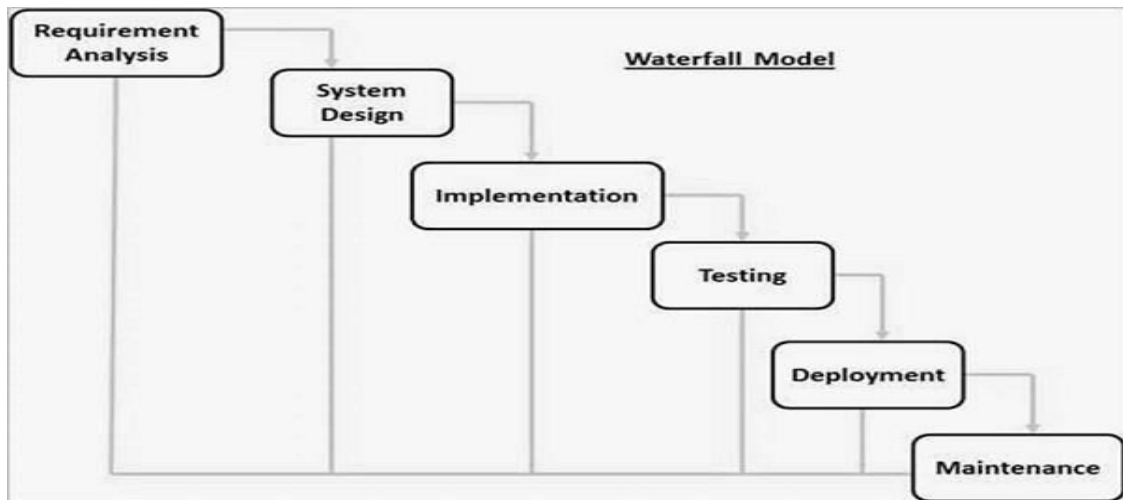
## **MODEL:**

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially

### **Waterfall Model - Design:**

In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. Following is the pictorial representation of Iterative and Incremental model:





The sequential phases in Waterfall model are –

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the

product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

### **Waterfall Model - Application:**

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are –

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

### **The advantages of the waterfall model SDLC Model are as follows:**

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Some of the major advantages of the Waterfall Model are as follows –

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.

- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

**The disadvantages of the waterfall model SDLC Model are as follows:**

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows –

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang" at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

## **CHAPTER II**

### **SYSTEM ANALYSIS**

## **Introduction**

System analysis is the process of collecting and interpreting facts, understanding problems and using the information to suggest improvement on the system. This will help to understand the existing system and determine how computers make its operation more effective. The aim of this analysis is to collect detailed information on the system and the feasibility study of the proposed system.

### **2.1 EXISTING SYSTEM**

In every Gym or Fitness Center there is no guarantee for a computer or system to keep record of the Customers to be present. There is no suitable timetable, customers workout is not arranged properly. So there can be a clash in daily schedule. Since the records are in written document, it is difficult to find the user names, Workout timings, summary etc. Administrator will be having a poor control over the Gym. They Using Paper work and direct human language communication to manage the Gym System will create problems, in terms of member records and their transactions which minimize the overall performance of the system and do not fulfill the requirements. The existing Gym Management System did not have a user-friendly interface. The details regarding gym members were manually written and recorded. There was no system of paying fees online. The gym members were not notified regarding the fee payment that were outstanding. Data redundancy and inconsistency makes the existing system odd and inefficient. Entering everything manually to the computer by creating a file is not exactly what we are talking about in computerization. The existing system requires a lot of manual work which results in taking more time than it should. The operations like updating and synchronizing data are also done manually in the existing system that is not automated and again time-consuming process. These practices are not at all reliable as the one wrong entry can take a lot of time in detection and then there is a correction. Humans are prone to errors and can mistakes often unless it has some inbuilt programs which can take check the input and save from error. We introduced the system to reduce the manual work effectively as there is the backend of the system which will take care of synchronizing and updating the data for the system.

#### **2.1.1 The Disadvantages of Existing System**

The existing system has the following disadvantages,

- ° They Use Paper work and direct human language communication to manage the Gym System
- ° There is no suitable timetable and customers work outs are not arranged properly.

## **2.2 PROPOSED SYSTEM**

Fitness Gym Management System provides a computer-based management system for keeping all records about Members, Machinery, Expenses, transactions and Salaries in an efficient and accessible database. This system helps the Owner and Admin to maintain large data about users and their daily transactions in gymnasium System is helping in creating reports and other record. The system is also suitable for users for an automated attendance and online profile Our Gymnasium Management System is the best option for it. It reduces and removes the manual and traditional workload, Administrator can easily add/ delete/ update/view each record on the computer. In the gym management system, after the planning and analysis phase of the system gets completed. Then the next phase required to transform the collected required system information into a structural blueprint which will serve as a reference while constructing the working system. It is a phase when most of the risks and errors unveiled so it's is good practice to take care of this thing from the start. This is a fully-fledged system that will be the backbone of the whole management of the gym so ignoring the risk or error is not an option as later it can make a greater form of itself. So, it is better to minimize the problems faced by both staff and the manager in the Organization. Another interesting part is that the users of the application can check by themselves their chance of having heart disease by themselves.

### **2.2.1 Advantages of Proposed System**

- It is a fully-fledged system that will be the backbone of the whole management of the gym.
- Can check the chance of having heart disease by themselves
- It reduces and removes the manual and traditional workload.
- Can be handled using a computer than manual checking.

## **2.3 Feasibility Study**

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioral feasibilities.

The proposed system is theoretically investigated to check the feasibility and found that they are more reliable and efficient in the cases given below. There are three aspects in the feasibility study portion of the preliminary investigation.

✓ Economic feasibility

✓ Technical feasibility

✓ Behavioural feasibility

The proposed system must be evaluated from a technical point of view first, and if technical feasible their impact on the organization must be assessed. If compatible, the operational system can be devised. Then they must be tested for economic feasibility.

### **2.3.1 Economic Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors which affect the development of a new system is the cost it would require. Since the system developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

### **2.3.2 Technical Feasibility**

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, procedures and staff. Having identified an outline system, the investigation must go on suggest the type of equipment, required method developing the system, of running the system once it has been designed. The project

should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Though the technology become obsolete after some period of time, due to the fact that newer version of some software supports older versions, the system still be used. So there are only minimal constraints involved with this project. The system has been developed using C# and .NET, along with the database software SQL server, thus we could conclude that the project is technically feasible for development.

### **2.3.3 Behavioural Feasibility**

People are inherently resistant to change and computers have been known to facilitate change. The System is designed in user friendly manner and we need to provide any special training for the persons using this software. The operating system used is Windows 11, which is also user friendly. Since the application is web biased and can easily accessed in a web browser, which is quite familiar to the intended users, it does not have any operational barriers. So no need to provide any special training for using this application software and hence it is behaviourally feasible.

## **2.4 System Specifications**

System Specification deals with the technical aspects the project has to meet in minimum to work successfully. This also includes the different aspects the software requirement is determined from. The technical details typically include:

- Software Specification
- Hardware Specification

### **2.4.1 Software Specifications**

The software required for the application depends on the following factors:

- ✓ The flexibility of the software
- ✓ Software contracts
- ✓ Limitation of the software

## **Software Requirement**

This specifies the minimum software requirements for implementing the system. This includes:



- Front End: - HTML, CSS, Java Script, Bootstrap
- Back End: - My SQL
- Client side scripting: HTML
- Server side scripting: Python
- Platform:-Flask
- Operating System: Microsoft windows 8

### **2.4.2 Hardware Specifications**

The software required for the application depends on the following factors:

- ✓ Determining size and capacity requirements.
- ✓ Computer evaluation and measurement.
- ✓ Financial factors.
- ✓ Maintenance and support.

### **Hardware Requirement**

- Microprocessor: Any 64 bit processor.
- Clock speed: - 2.13GHz
- Ram: 1 GB and above
- Hard disk: 40 GB and above
- Keyboard:-standard keyboard
- Mouse: Standard mouse
- Connectivity: - LAN & Wi-Fi
- Camera: Standard Camera

## **2.5 Identification of Actors**

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yields an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

We can identify the actors through a list of questions. The answers to these questions bring out the actors of the system is.

- Admin

- Gym Instructor
- Physician
- User
- Doctor

Here we need to specify the use cases of each actor.

## 2.6 Identification of use cases

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yield an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

To find out the use cases, ask the following questions to each of the actors.

- ✓ Which functions does the actor require from the system? What does the actor need to do?
- ✓ Does the actor need to read, create, destroy, modify or store some kind of information in the system?
- ✓ Does the actor have to calculate something? And want to provide information for others?
- ✓ Could the actor's daily work be simplified or made more efficient by adding new functions to the system (typically functions which are currently not automated in the system)?

### 2.6.1 Use cases for the actor Admin

- Login
- Register employee
- Register physician
- Verify user
- Add gym requirements and update
- View complaints
- Add new batches
- Allot user to gym instructor
- Add new completion details

- View best performer
- View feedback
- Prefer best performer to the competition
- View attendance of user
- Doctor Management.
- a Approve Doctor
- b. View Approved Doctors
- View doctor reviews
- Video Management (Tutorial)

#### **2.6.2 Use cases for the actor Physician**

- Login
- Store medicine info
- View users
- Update medicine info
- View doubts
- View bookings
- Booking history
- Payment history

#### **2.6.3 Use cases for the actor Gym Instructor**

- Add user details
- Add timing of batch
- View and update of user
- Add diet chart for user
- View medicine info
- Add attendance details
- view competition

#### **2.6.4 Use cases for the actor User**

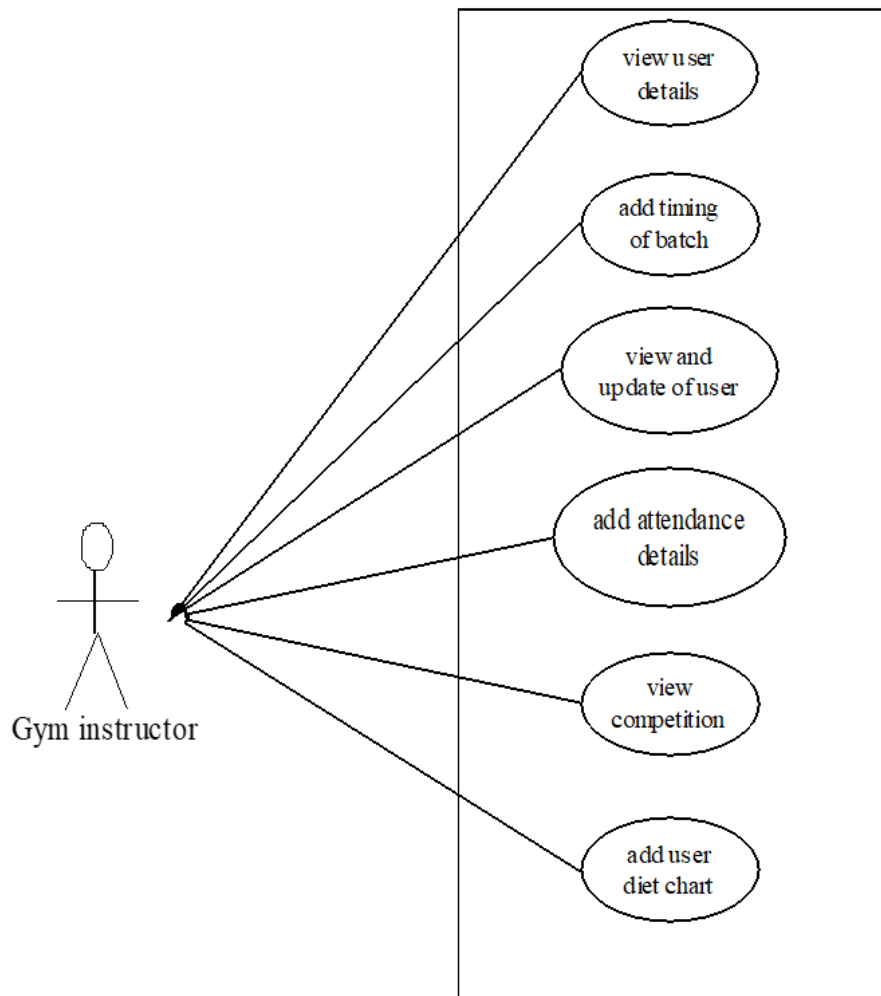
- Online registration by paying a certain amount as registration fee.
- Diet chart
- Drop gym
- View gym equipment

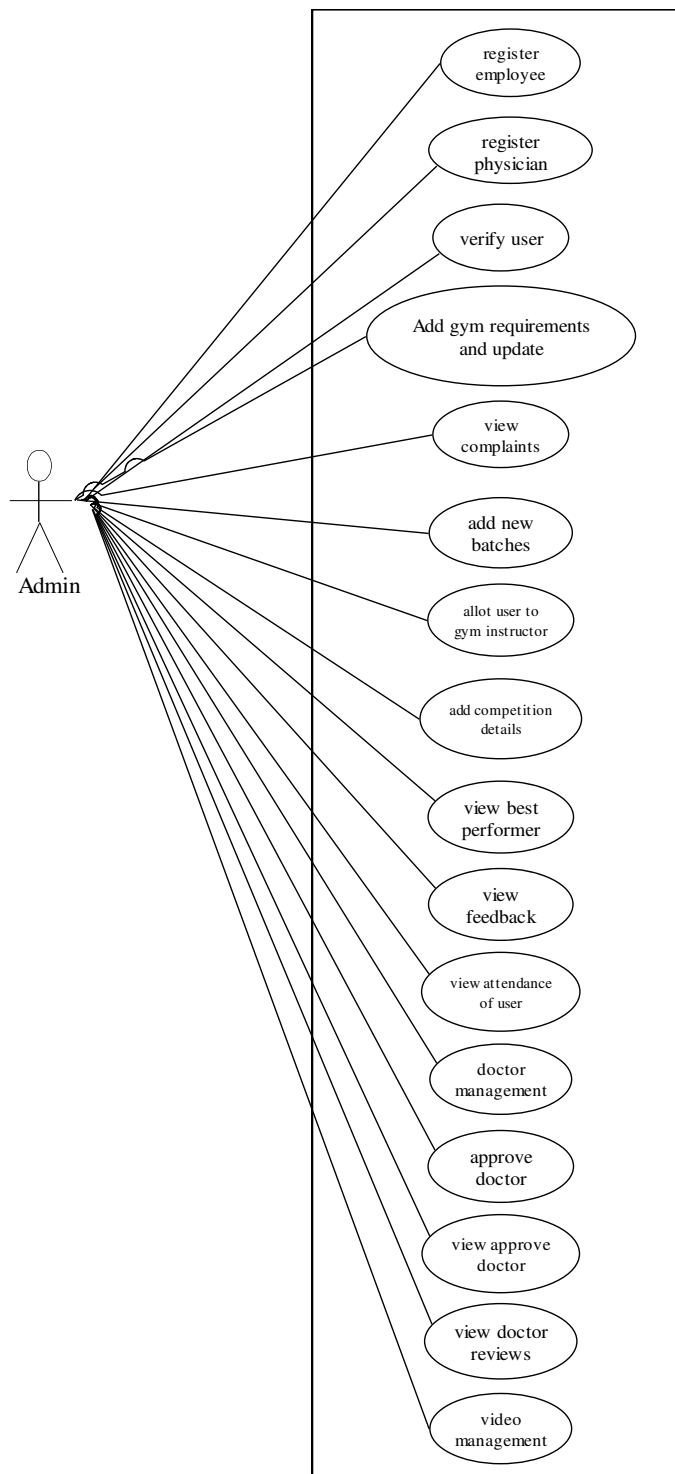
- Buy medicine
- Send doubts
- Send complaints
- Send feedback
- View completions details
- Pay monthly fee
- View Doctors
- View Doctors Schedule
- Make doctor's appointment
- Predict diseases using Symptoms and view result
- Send rating

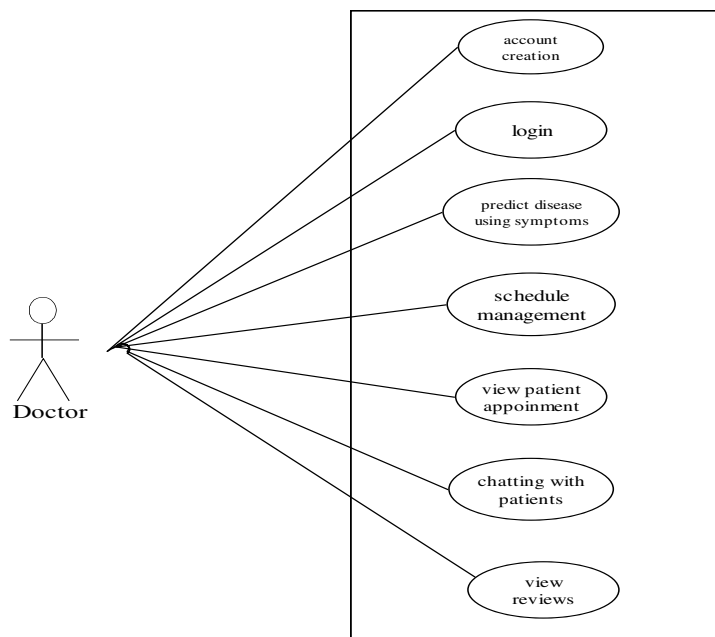
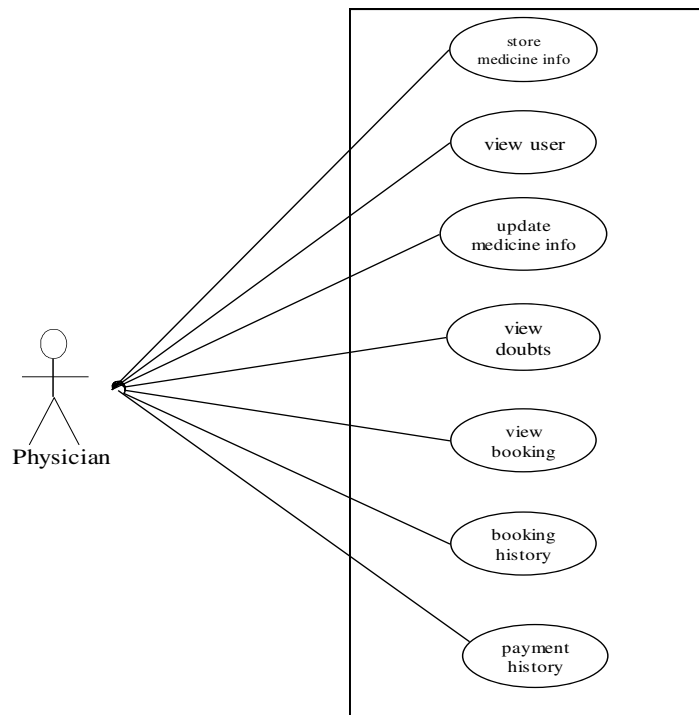
#### **2.6.5 Use cases for the actor Doctor**

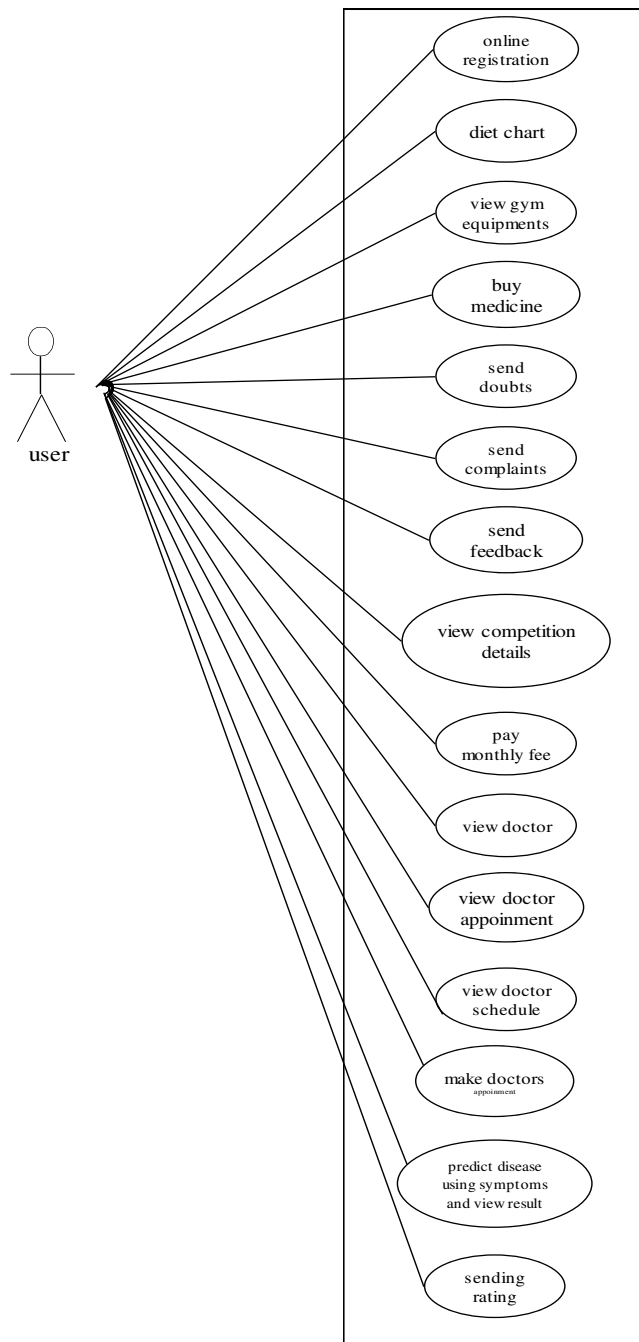
- Account Creation
- Login
- View Profile/Edit Profile/ Change Password
- Predict Diseases using Symptoms
- Schedule Management
- View Patients appointments
- Chatting with Patients

## 2.6.6 USE CASE DIAGRAM











## **CHAPTER III**

### **SYSTEM DESIGN**

### **3.1 INTRODUCTION:**

System design provides an understanding of the procedural details, necessary implementing the system recommended in the feasibility study. Basically it is all about the creation of a new system. This is a critical phase since it decides the quality of the system and has a major impact on the testing and implementation phases.

**System design consists of three major steps.**

- Drawing of the expanded system data flow charts to identify all the processing functions required.
- The allocation of the equipment and the software to be used.
- The identification of the test requirements for the system.

### **CHARACTERS OF DESIGN**

- A design should exhibit a hierarchical organization that makes intelligent use of control among components of the software.
- A design should be modular that is, the software should be logical.
- A design should contain distinct and separable representation of data and procedure.
- A design should lead to interface that reduce the complexity of the connections between modules and with the external environment.

### **3.2 Database Design**

A Database is a collection of inter related data stored with minimum redundancy to serve many users quickly and efficiently. In database design data independence, accuracy, privacy and security are given higher priority. Database design is an integrated approach to the file design. This activity deals with the design of the physical data base. All entities and attributes have been identified while creating the database. The database design deals with the grouping of data into number of tables so as to.

- ✓ Reduplication of data.
- ✓ Minimize storage space.
- ✓ Retrieve the data efficiently.

Following are some guidelines for the database design:

- Design a relational schema so that it is easy to explain its meaning. Do not combine attributes from multiple entity and relationship type into a single relation.
- Design the database schema so that no insertion, deletion or modification anomalies are present in the relation.
- As far as possible, avoid placing attributes in the base relation whose values may frequently be null.
- Design relation schema so that they can be joined with equality conditions on attributes that are either primary keys or foreign keys in a way that no spurious tuples are generated.

### 3.3 Table Design

DB design is required to manage large bodies of information. The management of data involves both the definition of the structure of storage of information and provisions of mechanism for the manipulation of information. For developing an efficient database certain conditions have to be fulfilled such as:

- Control Redundancy
- Ease of Use
- Data Independence
- Accuracy and Integrity

There are five major steps in design process:

- Identify the table and relationship.
- Identify the data that is needed for each table and relationship.
- Resolve the relationship.
- Verify the design.
- Implement the design

The Database Consist of the following tables given below.

### 3.4Bfit Ai-based fitness center

#### *1.Login table*

Colum name	Data type	Constraints	Description
login_id	Int	primary key	unique identifier
username	varchar(50)	not null	name of user
password	varchar(50)	not null	secret key
usertype	varchar(50)	not null	To specify the role

#### *2.Doctortable*

Colum name	Data type	Constraints	Description
doctor_id	Int	primary key	unique identifier
doctor_name	varchar(50)	not null	Name of doctor
experience	varchar(50)	not null	Experience of doctor
qualification	varchar(50)	not null	Qualification of doctor
email_id	Varchar(100)	Not null	Email id of doctor
phone_no	Varchar(100)	Not null	Phone no of doctor
Dob	Varchar(100)	Not null	Date of birth
gender	Varchar(100)	Not null	Gender of doctor

#### *3. Physiciantable*

Colum name	Data type	Constraints	Description
physician_id	Int	primary key	unique identifier
Name	varchar(50)	not null	Nameof physician
email_id	varchar(50)	not null	Email id of physician
phone	varchar(50)	not null	Phone no of physician
dob	varchar(50)	not null	Date of birth
gender	varchar(50)	not null	Gender of physician
qualification	varchar(50)	not null	Qualification of physician

#### 4. Batch table

Colum name	Data type	Constraints	Description
batch_id	Int	primary key	unique identifier
date_of_join	varchar(50)	not null	Date of join
time_from	varchar(50)	not null	Time from
time_to	varchar(50)	not null	Time to
batch_name	varchar(50)	not null	Batch name

#### 5. Employee table

Colum name	Data type	Constraints	Description
employee_id	Int	primary key	unique identifier
employee_name	varchar(50)	not null	Name of employee
email_id	varchar(50)	not null	Email id of employee
phone_no	varchar(50)	not null	Phone no of employee
Dob	varchar(50)	not null	Date of birth
gender	varchar(50)	not null	Gender of employee
experience	varchar(50)	not null	Experience of employee

#### 6. User table

Colum name	Data type	Constraints	Description
user_id	Int	primary key	unique identifier
name	varchar(50)	not null	Name of user
email_id	varchar(50)	not null	Email id of user
Phone_no	varchar(50)	not null	Phone no of user
dob	varchar(50)	not null	Date of birth
gender	varchar(50)	not null	Gender of user
weight	varchar(50)	not null	Weight of user
height	varchar(50)	not null	Height of user
bmi	varchar(50)	not null	Body mass index

### 7. Scheduletable

Colum name	Data type	Constraints	Description
schedule_id	Int	primary key	unique identifier
doctor_id	Int	not null	Doctor id
date	varchar(50)	not null	Date of schedule
time_from	varchar(50)	not null	Time of schedule
time_to	varchar(50)	not null	Time to schedule

### 8. Stock table

Colum name	Data type	Constraints	Description
stock_id	Int	primary key	unique identifier
physician_id	Int	not null	Physician id
medicine_id	Int	not null	Medicine id
stock	varchar(50)	not null	Stock details

### 9. Performer table

Colum name	Data type	Constraints	Description
performer_id	Int	Primary key	unique identifier
applicant_id	Int	not null	Applicant id

### 10. Review table

Colum name	Data type	Constraints	Description
review_id	Int	Primary key	unique identifier
date	Varchar(50)	Not null	Date of review
user_id	Int	Not null	User id
doctor_id	Int	Not null	Doctor id
review	Varchar(50)	Not null	Review

*11. Payment table*

Colum name	Data type	Constraints	Description
payment_id	Int	Primary key	unique identifier
date	Varchar(50)	Not null	Date of payment
booking_id	Int	Not null	Booking id
time	Varchar(50)	Not null	Time of payment
account_no	Varchar(50)	Not null	Account number

*12. Medicine table*

Colum name	Data type	Constraints	Description
medicine_id	Int	Primary key	unique identifier
medicine_name	Varchar(50)	Not null	Name of medicine
medicine_price	Varchar(50)	Not null	Price of medicine
description	Varchar(50)	Not null	Description for medicine

*13. Feedback table*

Colum name	Data type	Constraints	Description
feedback_id	Int	Primary key	unique identifier
user_id	Int	Not null	User id
date	Varchar(50)	Not null	Date of feedback
feedback	Varchar(50)	Not null	Feedback

*14. Equipment\_details table*

Colum name	Data type	Constraints	Description
equipment_id	Int	Primary key	unique identifier
equipment_name	Varchar(50)	Not null	Name of equipment
details	Varchar(50)	Not null	Details of equipments
photo	Varchar(50)	Not null	Photo of equipments

*15. Doubt table*

Colum name	Data type	Constraints	Description
doubt_id	Int	Primary key	unique identifier
user_id	Int	Not null	User id
date	Varchar(50)	Not null	Date of doubt
doubt	Varchar(50)	Not null	Doubts
reply	Varchar(50)	Not null	Reply for doubts

*16. Diet table*

Colum name	Data type	Constraints	Description
diet_id	Int	primary key	unique identifier
user_id	varchar(50)	not null	User id
break_fast	varchar(50)	not null	Break fast
lunch	varchar(50)	not null	Lunch
post_workout_food	varchar(50)	not null	Post workout food
pre_workout_food	varchar(50)	not null	Pre workout food
amount_of_protien	varchar(50)	not null	Amount of protien
calorie_of_food	varchar(50)	not null	Calorie of food

*17. Complaint table*

Colum name	Data type	Constraints	Description
complaint_id	Int	Primary key	unique identifier
user_id	Int	Not null	User id
date	Varchar(50)	Not null	Date of complaint
complaint	Varchar(50)	Not null	Complaint
reply	Varchar(50)	Not null	Reply for complaint



*18. Competition table*

Colum name	Data type	Constraints	Description
competition_id	Int	Primary key	unique identifier
competition_name	Varchar(50)	Not null	Name of competition
date	Varchar(50)	Not null	Date of competition
details	Varchar(50)	Not null	Details of competition

*19. Booking table*

Colum name	Data type	Constraints	Description
booking_id	Int	Primary key	unique identifier
user_id	Int	Not null	User id
physician_id	Int	Not null	Physician id
date	Varchar(50)	Not null	Date of booking
amount	Varchar(50)	Not null	Amount
status	Varchar(50)	Not null	Status of booking

*20. Bank table*

Colum name	Data type	Constraints	Description
bank_id	Int	Primary key	unique identifier
bank_name	Varchar(50)	Not null	Bank name
account_no	Varchar(50)	Not null	Account number
ifsc	Varchar(50)	Not null	Ifsc code
balance	Varchar(50)	Not null	Balance amount

*21. Attendance table*

Colum name	Data type	Constraints	Description
attendance_id	Int	Primary key	unique identifier
user_id	Int	Not null	User id
date	Varchar(50)	Not null	Date
check_in	Varchar(50)	Not null	Check in
check_out	Varchar(50)	Not null	Check out

### 22.Appointment table

Colum name	Data type	Constraints	Description
appointment_id	Int	Primary key	unique identifier
user_id	Int	Not null	User id
schedule_id	Int	Not null	Schedule id
date	Varchar(50)	Not null	Date of appointment
token_no	Varchar(50)	Not null	Token number

### 23.Allocation table

Colum name	Data type	Constraints	Description
allocation_id	Int	Primary key	unique identifier
user_id	Int	Not null	User id
batch_id	Int	Not null	Batch id
instructor_id	Int	Not null	Instructor id

## 3.5. Data Flow Diagram

A graphical representation is used to describe and analyses the movement of data through a system manual or automated including the processes, Storing of data and delays in the system. Data flow diagrams are the central tool and the basis from which other components are developed.

The transformation of data, from input to output through process may be described logically and independently of the physical components associated with the system.

They are termed logical dataflow diagrams, showing the actual implementations and the movement of data between people, departments and workstations. DFD is one of the most important modelling tools used insystem design. DFD shows the flow of data through different process in the system.

### **PURPOSE:**

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have reasonably completed model

of the behavior of the system. It is important to avoid premature designs, wherein develop designs before analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

Throughout my project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In my opinion, "efficient design of the data flow and context flow diagram helps to design the system successfully without much major flaws within the scheduled time". This is the most complicated part in a project. In the designing process, my project took more than the activities in the software lifecycle. If we design a system efficiently with all the future enhancements the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as a way of expressing system requirements in graphical form. A data flow diagram also known as "bubble chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It functionality decomposes the requirement specification down to the lowest level. Data Flow Diagram depicts the information flow, the transformation flow and the transformations that are applied as data move from input to output. Thus DFD describes what data flows rather than how they are processed.

Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is one of the most important tools used during system analysis. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

Data Flow Diagrams are made up of a number of symbols, which represents system components. Data flow modelling method uses four kinds of symbols, which are used to represent four kinds of system components.

These are

- Process
- Data stores
- Data flows

- External entity

**Process:**

Process shows the work of the system. Each process has one or more data inputs and produce one or more data outputs. Processes are represented by rounded rectangles in Data Flow Diagram. Each process has a unique name and number. This name and number appears inside the rectangle that represents the process in a Data Flow Diagram.

**Data Stores:**

A data stores is a repository of data. Processes can enter data, into a store or retrieve the data from the data store. Each data has a unique name.

**Data Flows:**

Data flows show the passage of data in the system and are represented by lines joining system components. An arrow indicates the direction of flow and the line is labelled by name of the dataflow.

**External Entity:**

External entities are outside the system but they either supply input data into the system or use other systems output. They are entities on which the designer has control. They may be an organizations customer or other bodies with which the system interacts. External entities that supply data into the system are sometimes called source. External entities that use the system data are sometimes called sinks. These are represented by rectangles in the

Data Flow Diagram.

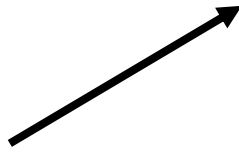
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Basic data flow diagram symbols are.....

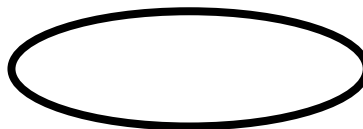
- A Square defines a source (originator) or destination of a system data:



- An Arrow identifies data flow. It is a pipeline through which information flows:



- A Circle represents a process that transforms incoming data flow(s) into outgoing data flow(s):



- An Open Rectangle is a data store:

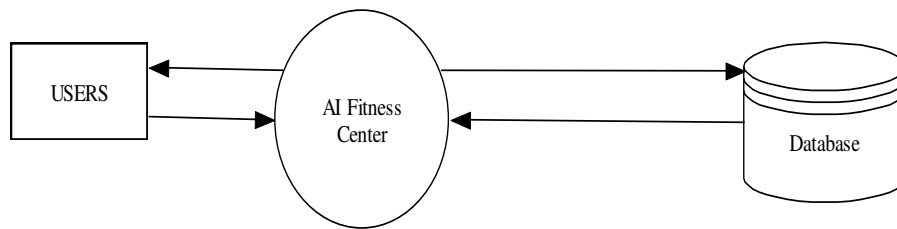


**Four steps are commonly used to construct a DFD:**

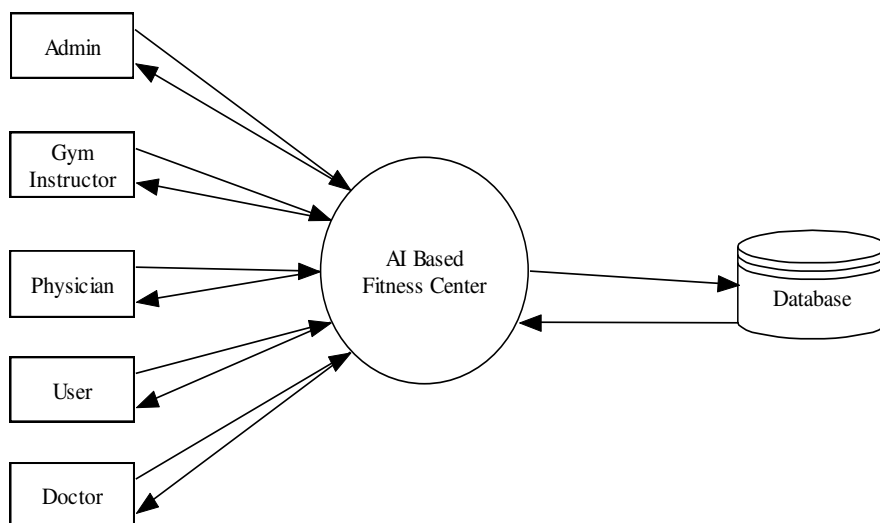
- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and left to right.
- When a process is exploded in to lower level details they are numbered.
- The names of data stores, sources and destinations are written in Capital letters.

## *DFD Level-0*

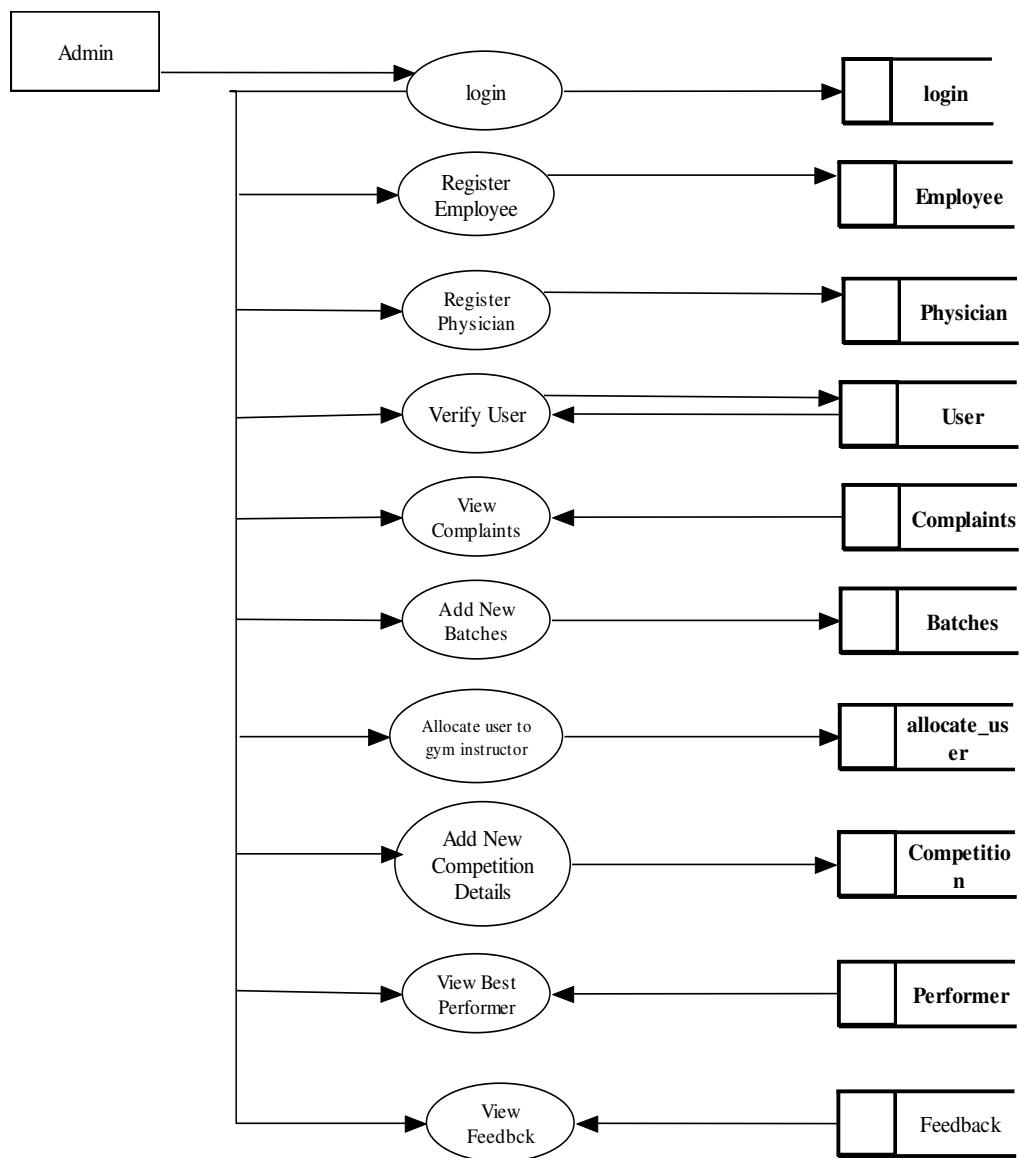
LEVEL 0



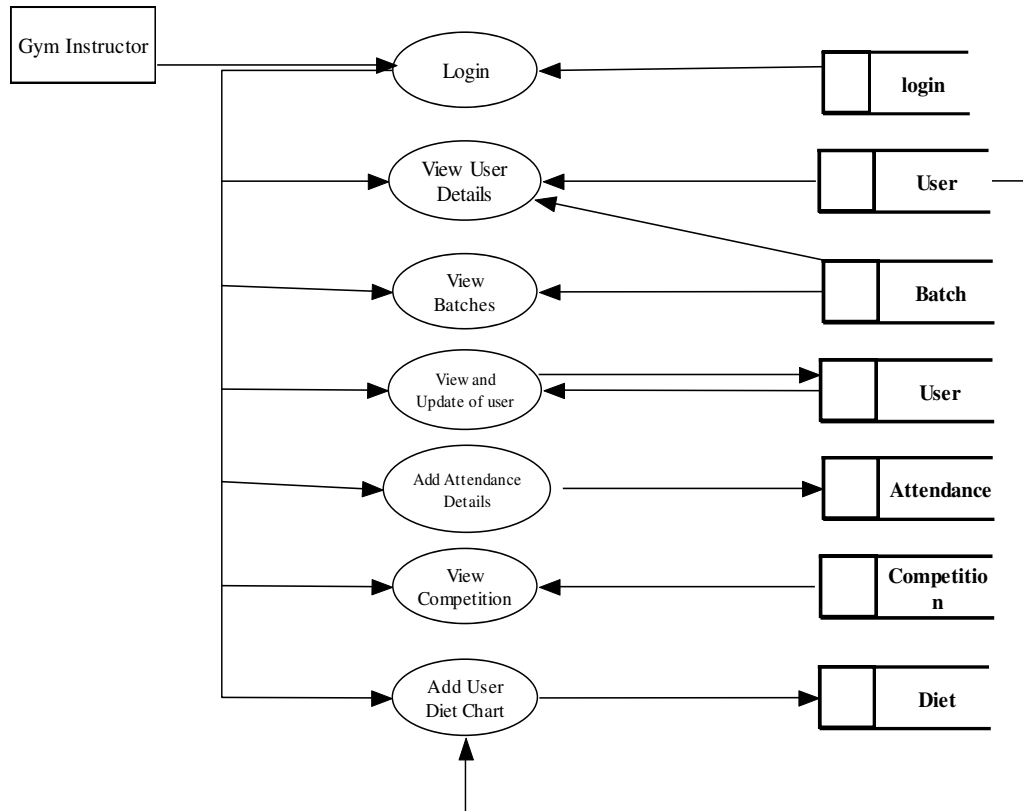
## *DFD Level-1*



*DFD Level-1.1ADMIN*

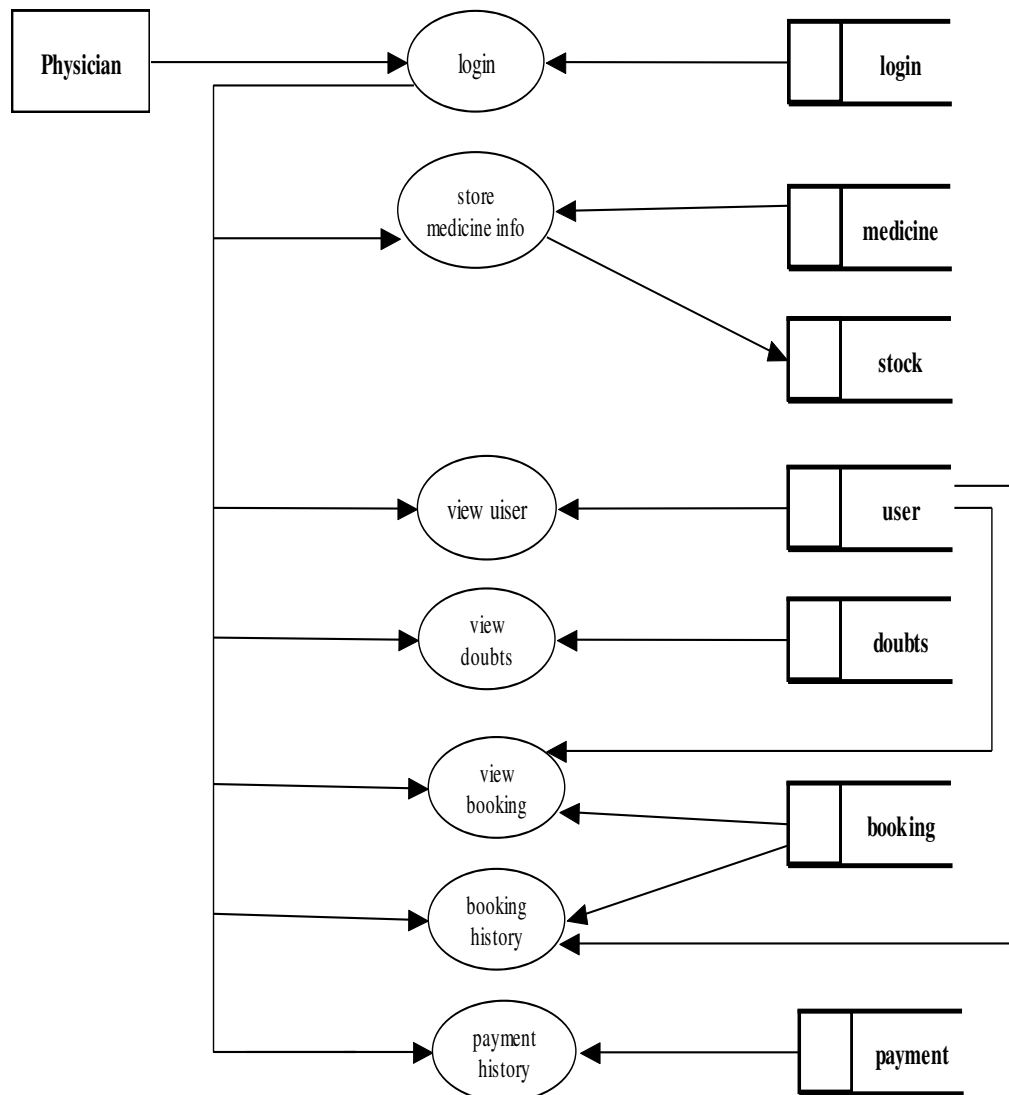


*DFD Level-1.2GYM INSTRUCTOR*

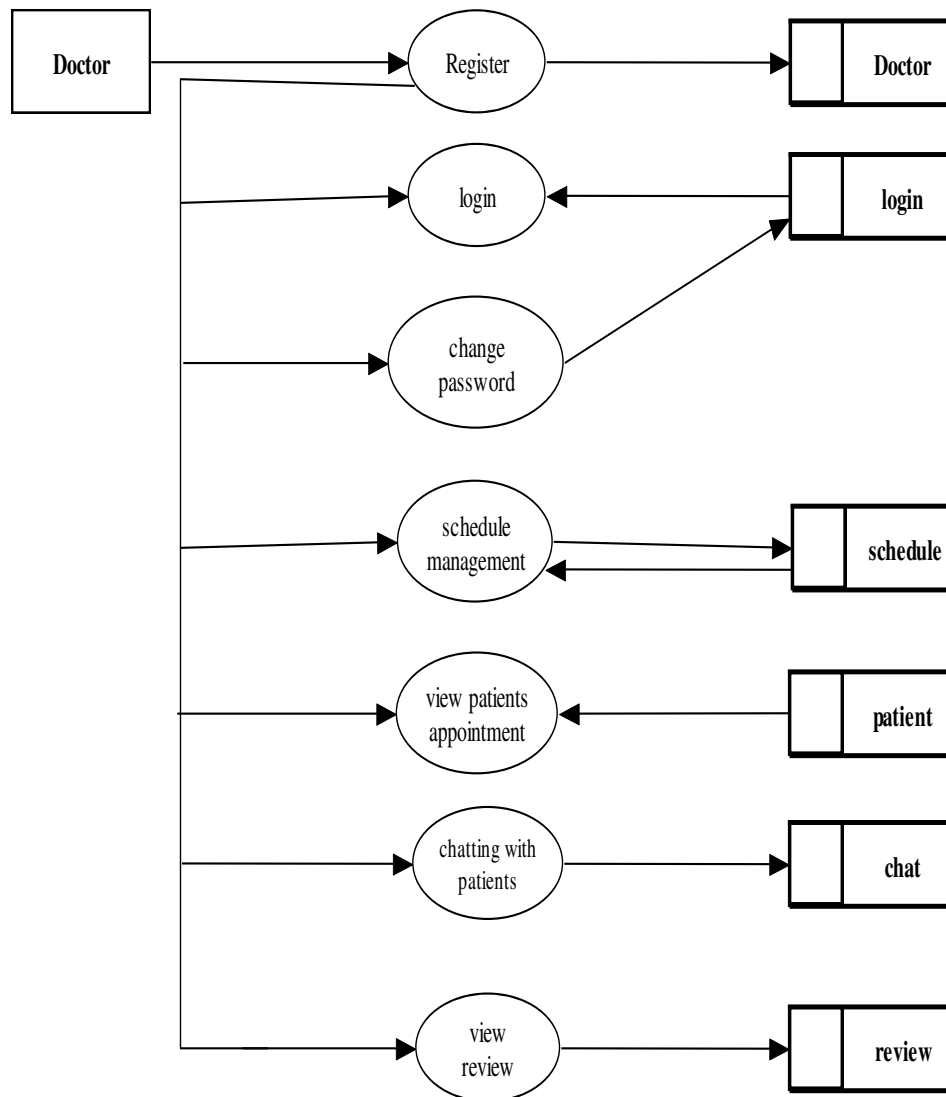




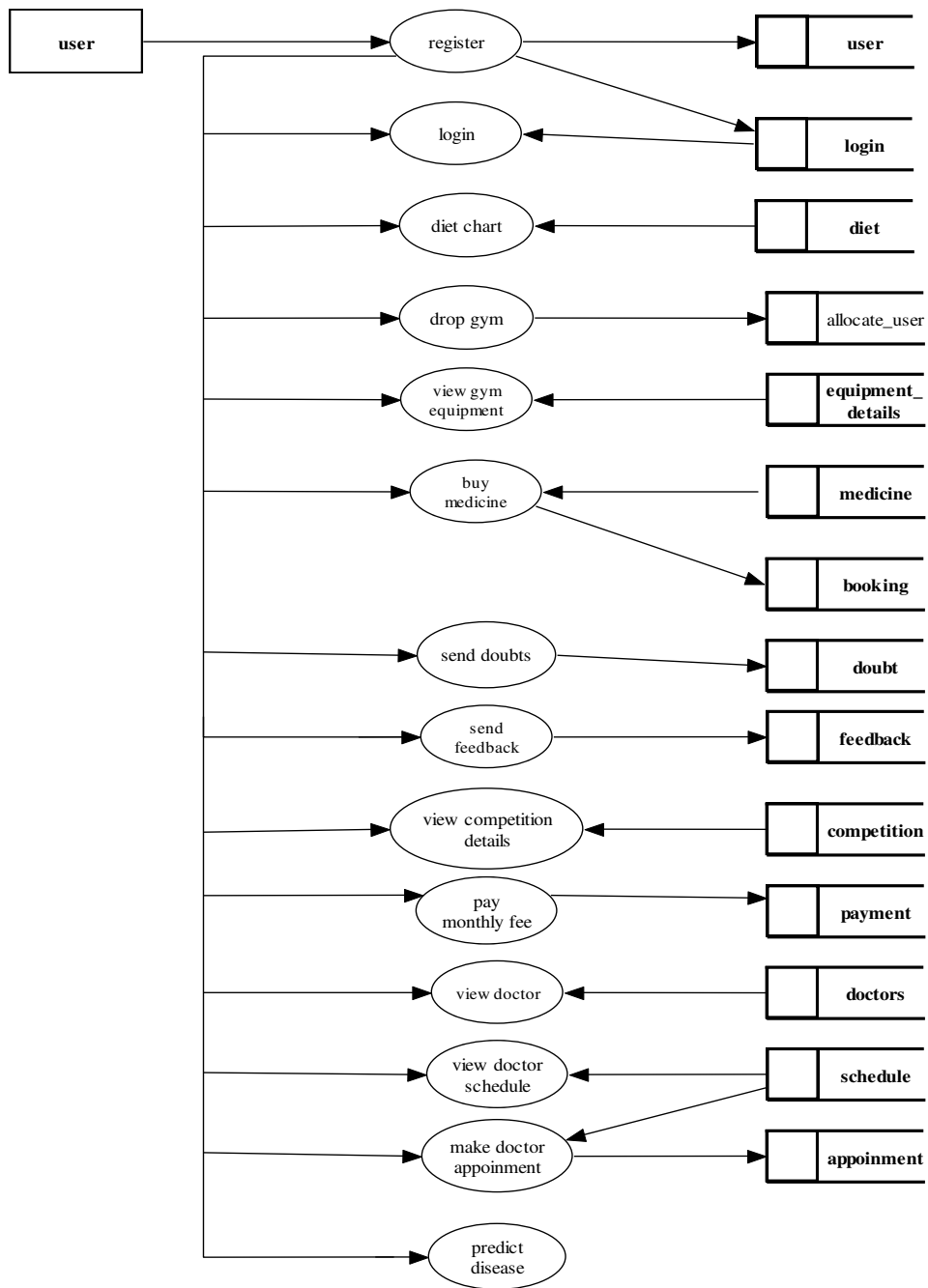
*DFD Level-1.3PHYSICIAN*



*DFD Level-1.4DOCTOR*



# DFD Level-1.5USER



### 3.6 ER Diagram

An ER diagram is a diagram that helps to design databases in an efficient way. It is a data model for describing the data or information. It is a visual representation of data that describes how data is related to each other. The main components of ER models are entities, attributes and the relationships that can exist among them.

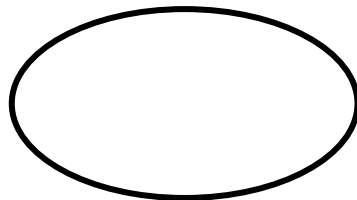
#### Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.



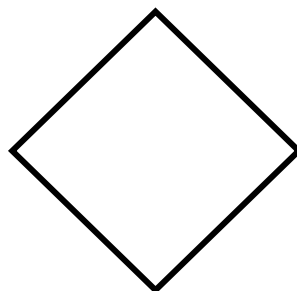
#### Attribute

Attributes are properties of entities. Attributes are represented by means of eclipses. Every eclipse represents one attribute and is directly connected to its entity (rectangle).

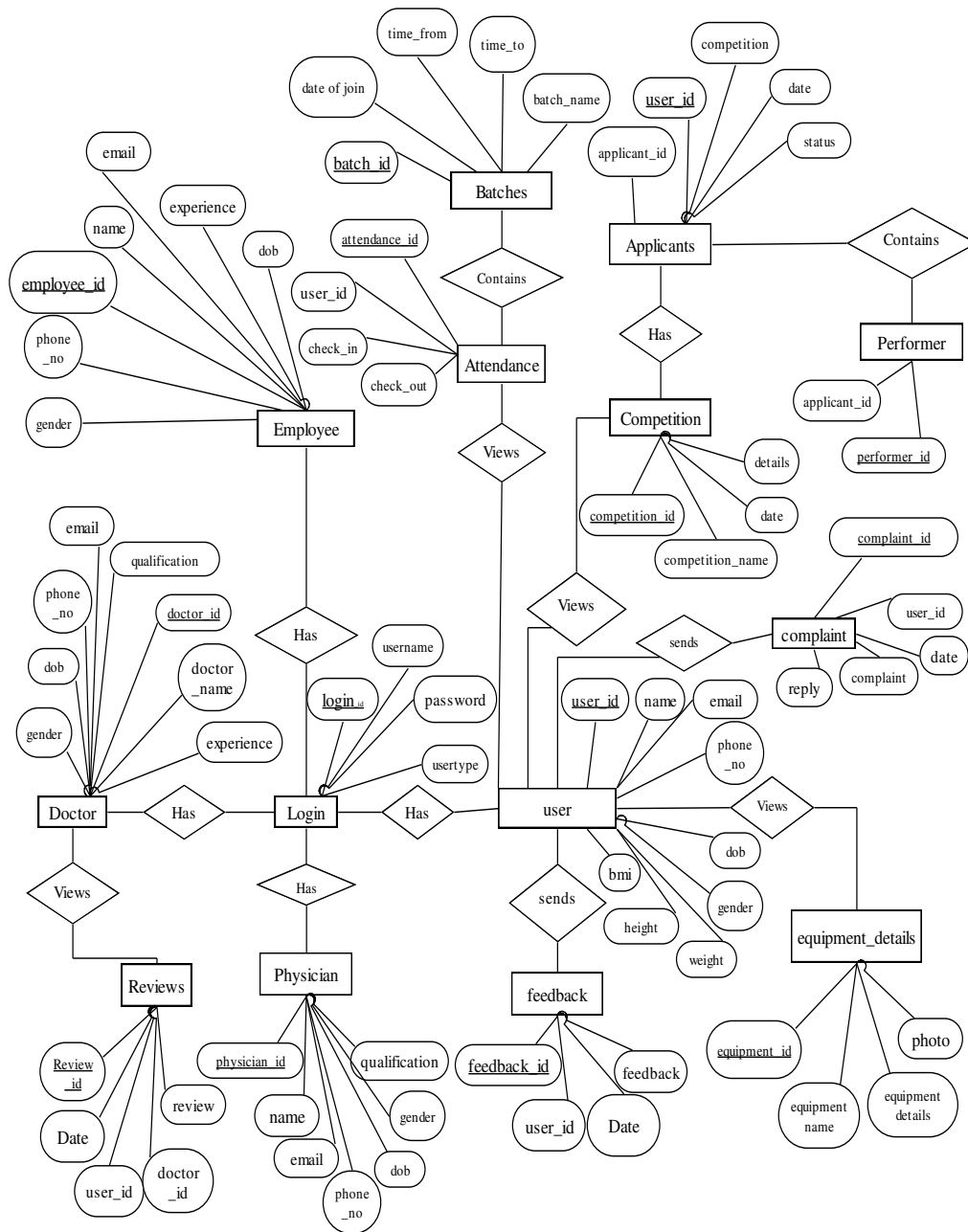


#### Relationship

Relationships are represented by diamond shaped box. Name of the relationship is written in the diamond box. All entities (rectangles), participating in relationship, are connected to it by a line.



## Architectural design



## **CHAPTER IV**

### **CODING**

## **4.1 INPUT INTERFACE**

Input design is a part of overall system design, which requires very careful attention. If data going into the system is correct, then the processing and output will magnify these errors. Thus the designer has a number of clear objectives in the different stages of input design.

- To produce a cost effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that input is acceptable to and understand by the user.

## **4.2 OUTPUT INTERFACE**

At the beginning of the output design various types of outputs such as external, internal, operational and interactive and turn around are defined. Then the format, content, location, frequency, volume and sequence of the outputs are specified. The content of the output must be defined in detail. The system analysis has two specific objectives at this stage.

- To interpret and communicate the results of the computer part of a system to the users in a form, which they can understand, and which meets their requirements.
- To communicate the output design specifications to programmers in a way in which it is unambiguous, comprehensive and capable of being translated into a programming language.

## **4.3 SOFTWARE DESCRIPTION**

### **4.3.1 HTML**

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML

specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

HTML files are written in ASCII text, so the user can use any text editor to create his/her web page, though a browser of one sort or another is necessary to view the web page. HTML is case insensitive with its language commands. The characters within the document, however, are case sensitive. The language consists of various "tags" which are known as elements. These allow the browser to understand (and put into the desired/specified format) the layout, background, headings, titles, lists, text and/or graphics on the page. The elements are classified according to their function in the HTML document. There are head elements and body elements. The head elements identify properties of the entire document, while body elements actually mark text as content and show a change in the appearance in one way or another. Most elements have a beginning and an ending which encompass the text the user wishes to mark with the tag. All HTML documents must begin with the element and end with the element. Some of the other elements which may be used are tags to create lists-- both ordered lists as well as unordered lists. The user may also create larger or smaller, bolder, italicized, or underlined text. Attributes may be used along with the elements. These perform functions such as placement of text, indication of the source files of images, and identification of links to the document or part of the document.

#### **4.3.2 CSS**

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging web pages, user interfaces for web applications, and user interfaces for many mobile applications. CSS is designed primarily to enable the separation of document content from document presentation, including aspects such as the layout, colours, and fonts.



## **Advantages of CSS**

- CSS saves time – you can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
- Pages load faster – If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
- Easy maintenance – To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- Superior styles to HTML – CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- Multiple Device Compatibility – Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- Global web standards – Now HTML attributes are being deprecated and it is being recommended to use CSS. So it is a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.
- Offline Browsing – CSS can store web applications locally with the help of an offline cache. Using of this, we can view offline websites. The cache also ensures faster loading and better overall performance of the website.
- Platform Independence – The Script offer consistent platform independence and can support latest browsers as well.

### **4.3.3 JAVASCRIPT**

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities. JavaScript was first known as Live Script, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name Live Script.

The General-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

Advantages of JavaScript:

- Less server interaction – you can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.
- Immediate feedback to the visitors – They don't have to wait for a page reload to see if they have forgotten to enter something.
- Increased interactivity – you can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.
- Richer interfaces – you can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

#### **4.3.4 MySQL**

MySQL is an open-source relational database management system (RDBMS). MySQL is released under an open-source license. So you have nothing to pay to use it. MySQL is a very powerful program in its own right.

It handles a large subset of the functionality of the most expensive and powerful database packages. It uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.

It works very quickly and works well even with large data sets. It is very friendly to PHP, the most appreciated language for web development. It supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB). It is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

#### **Major features as available in MySQL 5.6**

- A broad subset of ANSI SQL 99, as well as extensions.
- Cross-platform support.

- Stored procedures, using a procedural language that closely adheres to SQL/PSM.
- Triggers.
- Cursors.
- Updatable views.
- Online DDL when using the InnoDB Storage Engine.
- Information schema.
- Performance Schema that collects and aggregates statistics about server execution and query performance for monitoring purposes.
- A set of SQL Mode options to control runtime behaviour, including a strict mode to better adhere to SQL standards.
- X/Open XA distributed transaction processing (DTP) support; two phase commit as part of this, using the default InnoDB storage engine.
- Transactions with savepoints when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
- ACID compliance when using InnoDB and NDB Cluster Storage Engines.
- SSL support → Query caching → Sub-SELECTs (i.e. nested SELECTs) .
- Built-in replication support (i.e., master-master replication and master-slave replication) with one master per slave, many slaves per master.
- Multi-master replication is provided in MySQL Cluster, and multimaster support can be added to unclustered configurations using Galera Cluster.
- Full-text indexing and searching.
- Embedded database library.
- Unicode support.
- Partitioned tables with pruning of partitions in optimizer.
- Shared-nothing clustering through MySQL Cluster.
- Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.
- Native storage engines InnoDB, MyISAM, Merge, Memory (heap), Federated, Archive, CSV, Blackhole, NDB Cluster.
- Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

## Advantages

MySQL database server has lots of advantages over its competitors. Some of these advantages have been explained below.

- Open Source and Cost Effective:

The best thing about MySQL server is that this is open source and it has a free version as well. By open source software, we mean that the code of the software is available and anyone can tailor it according to his requirement. Companies prefer MySQL because they don't have to pay anything for this excellent product.

- Portability:

MySQL is cross platform database server. MySQL can be run on a variety of platforms including Windows, OS2, Linux and Solaris. Portability of MySQL server makes it suitable for applications that target multiple platforms particularly web application. MySQL contains API for almost all the major programming languages and can be easily integrated with the languages like PHP, C++, Perl, C, Python and ruby. In fact, MySQL is a part of the famous LAMP (Linux Apache MySQL PHP) server stack which is used worldwide for web application development.

- Seamless Connectivity:

Various secure and seamless connection mechanisms are available in order to connect with MySQL server. These connections include named pipes, TCP/IP sockets and UNIX Sockets.

- Rapid Development and Continuous Updates:

Being an open source product, MySQL has a very large developer community which releases regular patches and updates for MySQL. Several database templates have been developed which can be readily used and modified resulting in rapid application development.

- Security:

MySQL server databases are extremely secure and all the data access scenarios are protected via password and good thing about these passwords is that they are stored in encrypted form and it is not easy to break these advanced and complex encryption algorithms.

### 4.3.5 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

Major features of python

- Easy to code
- Free and Open Source
- Object-Oriented Language
- GUI Programming Support
- High-Level Language
- Extensible feature
- Python is **Portable** language
- Python is Integrated language

## **Advantages**

- Extensive support libraries
- Integration feature
- Improved productivity
- Easy to learn and write
- Vast library support
- Free and open source

### **4.3.6 Flask**

Flask is a web application framework written in Python. It was developed by Armin Ronacher, who led a team of international Python enthusiasts called Poocco. Flask is based on the Werkzeug WSGI toolkit and the Jinja2 template engine. Both are Poocco projects.

It is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier. It gives developers flexibility and is a more accessible framework for new developers since you can build a web application quickly using only a single Python file. Flask is also extensible and doesn't force a particular directory structure or require complicated boilerplate code before getting started.

## **CHAPTER V**

### **CODING PAGES**

## 5.1 Admin Page

```
@app.route('/')
def index():
    return render_template("index.html")

@app.route('/login')
def login():
    return render_template("login.html")

@app.route('/logout')
def logout():
    session.clear()
    session['lg']=" "
    return redirect('/')

@app.route('/login_post', methods=['post'])
def login_post():
    u=request.form['textfield']
    p=request.form['textfield2']
    db=Db()
    res=db.selectOne("select * from login where username='"+u+"'and
password='"+p+"'")
    if res is not None:
        session['lid']=res['login_id']
    if res['usertype']=='admin':
        session['lg']='lin'
    return redirect('/admin_home')

elif res['usertype'] == 'instructor':
    session['lg']='lin'

return redirect('/gyminstructor_home')

elif res['usertype'] == 'physician':
    session['lg']='lin'

session['lid'] = res['login_id']

return redirect('/physician_home')

elif res['usertype'] == 'doctor':
    session['lid'] = res['login_id']
```



```

        session['lg']='lin'

    return redirect('/doctor_home')

    elif res['usertype'] == 'user':
        session['lid'] = res['login_id']
        session['lg']='lin'
    return redirect('/user_home')
    else:
    return "invalid"
    else:
    return "invalid"

@app.route('/admin_home')
def admin_home():
    if session['lg']!='lin':
    return redirect('/')
    return render_template("admin/index.html")

@app.route('/add_batch')
def add_batch():
    if session['lg'] != 'lin':
    return redirect('/')
    return render_template("admin/add_batches.html")

@app.route('/add_batch_post',methods=['post'])
def add_batch_post():
    if session['lg'] != 'lin':
    return redirect('/')
    dj=request.form['textfield']
    tf=request.form['textfield2']
    tt=request.form['textfield3']
    bn=request.form['textfield4']
    db=Db()
    db.insert("insert into batches
VALUES('','+dj+', '"+tf+', '"+tt+', '"+bn+')")
    return '<script>alert("success");window.location="/add_batch"</script>'
@app.route('/add_competition')
def add_competition():
    if session['lg'] != 'lin':
    return redirect('/')
    return render_template("admin/add_competition.html")

@app.route('/add_competition_post',methods=['post'])

```

```

def add_competition_post():
    if session['lg'] != 'lin':
        return redirect('/')
    cn=request.form['textfield']
    da=request.form['textfield2']
    de=request.form['textarea']
    db=Db()
    db.insert("insert into competition VALUES('','" + cn + "',''" + da + "',''" + de
    + "')")
    return '<script>alert("success");window.location="/add_competition"</script>'

@app.route('/add_employee')
def add_employee():
    if session['lg'] != 'lin':
        return redirect('/')
    return render_template("admin/add_employee.html")

@app.route('/add_employee_post',methods=['post'])
def add_employee_post():
    if session['lg'] != 'lin':
        return redirect('/')
    n=request.form['textfield']
    e=request.form['textfield2']
    pn=request.form['textfield3']
    dob=request.form['textfield4']
    g=request.form['radio']
    ex=request.form['textarea']
    pswd=random.randint(1000, 9999)
    db=Db()
    lid=db.insert("insert into login(username, password, usertype)
    values('"+e+"', '"+str(pswd)+"', 'instructor')")
    db.insert("insert into employee
    VALUES('"+str(lid)+"', '"+n+"', '"+e+"', '"+pn+"', '"+dob+"', '"+g+"', '"+ex+"')")
    return '<script>alert("success");window.location="/add_employee"</script>'

@app.route('/add_physician')
def add_physician():
    if session['lg'] != 'lin':
        return redirect('/')
    return render_template("admin/Add_physician.html")

@app.route('/add_physician_post',methods=['post'])
def add_physician_post():
    if session['lg'] != 'lin':
        return redirect('/')

```

```

        n=request.form['textfield']
em=request.form['textfield2']
pn=request.form['textfield3']
        dob=request.form['textfield4']
        g=request.form['radio']
qu=request.form['textarea']
db=Db()
db.insert("insert into physician
VALUES('','"+n+"','"+em+"','"+pn+"','"+dob+"','"+g+"','"+qu+"')")
return '<script>alert("success");window.location="/add_physician"</script>'

@app.route('/gym_requirements')
def gym_requirements():
if session['lg'] != 'lin':
return redirect('/')
return render_template("admin/gym_requirements.html")

@app.route('/gym_requirements_post',methods=['post'])
def gym_requirements_post():
if session['lg'] != 'lin':
return redirect('/')
en=request.form['textfield']
        ed=request.form['textarea']
ph=request.files['fileField']
        dt=time.strftime("%Y%m%d-%H%M%S")
ph.save(static_path + "equipments\\" + dt + '.jpg')
        path="/static/equipments/" + dt + '.jpg'
db = Db()
db.insert( "insert into equipment_details VALUES('','" + en + "','"+ ed +
        "','"+ str(path) + "')")
return '<script>alert("success");window.location="/gym_requirements"</script>'

@app.route('/allocate_batch/<userid>')
def allocate_batch(userid):
if session['lg'] != 'lin':
return redirect('/')
db=Db()

        res=db.select("select * from batches")
        res1=db.select("select * from employee")
return
render_template("admin/allocate_batch.html",data=res,data2=res1,u=userid)

@app.route('/allocate_batch_post',methods=['post'])
def allocate_batch_post():

```

```

if session['lg'] != 'lin':
return redirect('/')
userid=request.form['uid']
    b=request.form['select']
i=request.form['select2']
db = Db()
qry=db.selectOne("select * from allocate_user where user_id='"+userid+"' and
batch_id='"+b+"' and instructor_id='"+i+"' ")
if qryis not None:
return '<script>alert("Already
Allocated");window.location="/view_user"</script>'
else:
db.insert("insert into allocate_user VALUES('','"+userid+"','" + b + "','" + i
+ "')")
return '<script>alert("success");window.location="/view_user"</script>'

@app.route('/approve_doctor')
def approve_doctor():
if session['lg'] != 'lin':
return redirect('/')
db=Db()
    res = db.select("select * from doctor,login where
doctor.doctor_id=login.login_id and login.usertype='pending' ")

return render_template("admin/approve_Doctor.html",data=res)

@app.route('/adapprove_doctor/<d>')
def adapprove_doctor(d):
if session['lg'] != 'lin':
return redirect('/')
db=Db()
db.update("update login set usertype='doctor' where login_id='"+d+"' ")
return '<script>alert("approved");window.location="/approve_doctor"</script>'
@app.route('/reject_doctor/<d>')
def reject_doctor(d):
if session['lg'] != 'lin':
return redirect('/')
db=Db()
db.delete("delete from login where login_id='"+d+"' ")
db.delete("delete from doctor where doctor_id='"+d+"' ")
return '<script>alert("rejected");window.location="/approve_doctor"</script>'

@app.route('/view_applicants')
def view_applicants():
if session['lg'] != 'lin':

```

```

return redirect('/')

db=Db()

res=db.select("select * from applicants,user,competition where
applicants.user_id = user.user_id and applicants.competition_id =
competition.competition_id")

return render_template("admin/view_applicants.html", data=res)

@app.route('/delete_applicants/<d>')
def delete_applicants(d):
if session['lg'] != 'lin':
return redirect('/')

db=Db()

db.delete("delete from applicants where applicant_id='"+d+"' ")
return '<script>alert("deleted");window.location="/view_applicants"</script>'

@app.route('/View_attendance')
def View_attendance():
if session['lg'] != 'lin':
return redirect('/')

db=Db()

res=db.select("select * from attendance,user where
attendance.user_id=user.user_id")
return render_template("admin/View_attendance.html", data=res)

@app.route('/delete_attendance/<d>')
def delete_attendance(d):
if session['lg'] != 'lin':
return redirect('/')

db=Db()

db.delete("delete from attendance where attendance_id='"+d+"' ")
return '<script>alert("deleted");window.location="/View_attendance"</script>'

@app.route('/view_batches')
def view_batches():
if session['lg'] != 'lin':
return redirect('/')

db=Db()

res=db.select("select * from batches")
return render_template("admin/view_batches.html", data=res)

@app.route('/delete_batches/<d>')
def delete_batches(d):
if session['lg'] != 'lin':
return redirect('/')

```

```

db=Db()
db.delete("delete from batches where batch_id='"+d+"' ")
return '<script>alert("deleted");window.location="/view_batches"</script>'

@app.route('/view_competition')
def view_competition():
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    res=db.select("select * from competition")
    return render_template("admin/view_competition.html", data=res)
@app.route('/delete_competition/<d>')
def delete_competition(d):
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    db.delete("delete from competition where competition_id='"+d+"' ")
    return '<script>alert("deleted");window.location="/view_competition"</script>'

@app.route('/view_complaints')
def view_complaints():
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    res=db.select("select * from complaints,user where complaints.user_id =
user.user_id ")
    return render_template("admin/view_complaints.html", data=res)
@app.route('/send_reply/<d>', methods=['get', 'post'])
def delete_complaints(d):
    if session['lg'] != 'lin':
        return redirect('/')
    if request.method=="POST":
        rep=request.form['textarea']
    db=Db()
    db.update("update complaints set reply='"+rep+"' where complaint_id='"+d+"'")
    return redirect("/view_complaints")
    else:
    return render_template("admin/send_reply.html")

@app.route('/view_doctor')
def view_doctor():
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    res=db.select("select * from doctor,login where

```

```

doctor.doctor_id=login.login_id and login.usertype='doctor'")
return render_template("admin/view_doctor.html", data=res)

@app.route('/view_doctor_review/<g>')
def view_doctor_review(g):
if session['lg'] != 'lin':
return redirect('/')
db=Db()
    res=db.select("select * from review,user,doctor where review.user_id =
user.user_id and review.doctor_id = doctor.doctor_id and
review.doctor_id='"+g+"' ")
return render_template("admin/view_doctor_review.html", data=res)
@app.route('/delete_doctor_review/<d>')
def delete_doctor_review(d):
if session['lg'] != 'lin':
return redirect('/')
db=Db()
db.delete("delete from review where review_id='"+d+"' ")
return
'<script>alert("deleted");window.location="/view_doctor_review"</script>'

@app.route('/view_employee')
def view_employee():
if session['lg'] != 'lin':
return redirect('/')
db=Db()
    res=db.select("select * from employee")
return render_template("admin/view_employee.html",data=res)
@app.route('/delete_employee/<d>')
def delete_employee(d):
if session['lg'] != 'lin':
return redirect('/')
db=Db()
db.delete("delete from employee where employee_id='"+d+"' ")
return '<script>alert("deleted");window.location="/view_employee"</script>'

@app.route('/View_feedback')
def View_feedback():
if session['lg'] != 'lin':
return redirect('/')

db=Db()
    res=db.select("select * from feedback,user where
feedback.user_id=user.user_id")
return render_template("admin/View_feedback.html", data=res)

```

```

@app.route('/delete_feedback/<d>')
def delete_feedback(d):
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    db.delete("delete from feedback where feedback_id='"+d+"' ")
    return '<script>alert("deleted");window.location="/View_feedback"</script>'

@app.route('/view_gym_requirements')
def view_gym_requirements():
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    res=db.select("select * from equipment_details")
    return render_template("admin/view_gym_requirement.html",data=res)
@app.route('/delete_gym_requirements/<d>')
def delete_gym_requirements(d):
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    db.delete("delete from equipment_details where equipment_id='"+d+"' ")
    return
    '<script>alert("deleted");window.location="/view_gym_requirements"</script>'

@app.route('/view_performer')
def view_performer():
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    res=db.select("select * from performer,applicants,competition,user where
performer.applicant_id = applicants.applicant_id and
applicants.competition_id=competition.competition_id and
applicants.user_id=user.user_id")
    return render_template("admin/view_performer.html", data=res)
@app.route('/delete_performer/<d>')
def delete_performer(d):
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    db.delete("delete from performer where performer_id='"+d+"' ")
    return '<script>alert("deleted");window.location="/view_performer"</script>'

```



```

@app.route('/view_physician')
def view_physician():
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    res=db.select("select * from physician")
    return render_template("admin/view_physician.html",data=res)
@app.route('/delete_physician/<d>')
def delete_physician(d):
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    db.delete("delete from physician where physician_id='"+d+"' ")
    return '<script>alert("deleted");window.location="/view_physician"</script>'

@app.route('/view_user')
def view_user():
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    res=db.select("select * from user,login where user.user_id=login.login_id
    and login.usertype='user' ")
    return render_template("admin/view_user.html",data=res)
@app.route('/approve_user/<d>')
def approve_user(d):
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    db.update("update login set usertype='user' where login_id='"+d+"' ")
    return '<script>alert("approved");window.location="/view_user"</script>'
@app.route('/reject_user/<d>')
def reject_user(d):
    if session['lg'] != 'lin':
        return redirect('/')
    db=Db()
    db.delete("delete from login where login_id='"+d+"' ")
    db.delete("delete from user where user_id='"+d+"' ")
    return '<script>alert("rejected");window.location="/view_user"</script>'

```

# **CHAPTER VI**

## **SYSTEM TESTING**

## 6.1 TESTING AND EVALUATION

Testing is a process of executing a program with the intent of finding an error. Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design and coding. Testing includes verifications of the basic logic of each program and verification that the entire system works properly. Testing demonstrates that software functions appear to be working according to specification. In addition, data collected as testing is conducted provides a good indication of software quality as a whole. The debugging process is the most unpredictable part of the testing process. Testing begins at the module level and works towards the integration of the entire computer-based system. Testing and debugging are different activities, but any testing strategy for software testing must accommodate low level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system function, against customer requirements. No testing is complete without verification and validation parts. The goals of verification and validation activities are to assess and improve the quality of work products generated during the development and modification of the software. There are two types of verification: life cycle verification and formal verification. Life cycle verification is the process of determining the degree to which the products of the given phase of the development cycle fulfil the specification established during the prior process. Formal verification is the rigorous mathematical demonstration that source code conforms to its specifications. Validation is a process of evaluating software at the end of the software development process to determine conformance with the requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The primary objectives, when we test software are the following:

- Testing is a process of executing with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.
- A successful test is one uncovers undiscovered errors.

Thus, testing plays a very critical role in determining the reliability and efficiency of the software and hence is a very important stage in software development.

Tests are to be conducted on the software to evaluate its performance under a number of conditions. Ideally, it should so at the level of each module and also when all of them are integrated to form the completed system. Software testing is done at different levels.

## **6.2 TESTING STRATEGIES**

A strategy for software testing integrates software test case design method in to a well-planned series of steps that result in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote a customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project progresses.

**The general characteristics of software testing strategies are:**

- Testing begins at the component level and works “outward” toward the integration of the entire computer system.
- Different testing techniques are appropriate at different points in time.

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements. A strategy must provide guidance for the practitioner and set of milestones for the manager. Because the step on the test strategy occurs at a time when deadline pressure begins to rise, progress must be measurable and problem must surface as early as possible.

The software team’s approach to testing is defining a plan that describes an overall strategy and a procedure that defines specific testing steps and tests that will be conducted. In the proposed system, if the administrator makes any attempt to login to the application without entering his password, then the system will not allow the user to login to the application.

## **6.3 TESTING TECHNIQUES**

The various testing techniques are given below:

### **6.3.1 WHITE BOX TESTING**

White-box testing is also called as glass-box testing, is a test case design method that goes to the control structure of the procedural design to derive test cases. Using white box testing methods, the software engineer can derive test cases that,

- ✓ Guarantee that all independent paths within a module have been exercised at Least once.
- ✓ Exercise all logical decision on their true and false sides.
- ✓ Execute all loops at their boundaries and within their operational sides.
- ✓ Exercise internal data structure to ensure their validity.

White box testing was successfully conducted on our system. All independent paths within a module have been executed at least once and all logical decisions have been exercised on their true and false sides.

### **6.3.2 BLACK BOX TESTING**

Black-box testing is also called as behavioural testing, focuses on the functional requirement of the software. It is a complimentary approach that is likely uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories.

- ✓ Incorrect or missing functions.
- ✓ Interface errors.
- ✓ Error on data structures or external database access.
- ✓ Behaviour or performance errors.
- ✓ Initialization and termination errors.

Black box testing was successfully conducted on your system. The system was divided into a number of modules and testing was conducted on each module. We have tested the system for incorrect or missing functions, interface and performance errors.

### **6.3.3 UNIT TESTING**

Unit testing comprises the set of tests performed by an individual programmer prior to the integration of the system. Testing removes residual bugs and improves the reliability of the system.

Testing allows the developer to find out the design faults if any, and enable correction if needed. Exhaustive unit testing has to be carried out to ensure the validity of the data. In order to successfully test the entire package, unit testing is carried out. Each module was tested as when it was developed. Thus it proved easier to conduct minute testing operation and correct them then and there.

### **6.3.4 INTEGRATION TESTING**

Bottom-up integration is the traditional strategy used to integrate the component of a software system into a functional whole. Bottom-up integration consists of unit testing, followed by subsystem testing and followed by testing of the entire system. Unit testing has the goal of discovering errors in the individual parts of the system.

Parts are tested in isolation from one another in an artificial environment known as “Test Harness”, which consists of driver programs and data necessary to exercise the modules. Unit testing should be as exhaustive as possible to ensure that each representative case handled by each module has been tested. Unit testing is eased by a system structure that is composed of small loosely coupled modules.

Both control and data interfaces must be tested. Large software system may require several levels of subsystem testing. Lower level subsystems are successively combined to form higher level subsystems. In most software systems, exhaustive testing of subsystem capabilities is not feasible due to the combination complexity of the module interfaces. Therefore, test cases must be carefully chosen to exercise the interfaces in the desired manner.

### **6.3.5 ACCEPTANCE TESTING**

Acceptance testing involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements. It is not unusual for two sets of acceptance test to be run, those developed by the quality group and those developed by the customer.

In addition to the functional and performance tests, stress tests are performed to determine the limitation of the system. For example, a compiler might be tested to determine the effect of the symbol table overflow, or real-time system might be tested to determine the effect of simultaneous arrival of numerous high priority interrupts.

#### **6.3.6 OUTPUT TESTING**

Output testing of the proposed system is important since no system could be useful if it does not produce the required output. The output format on the screen is found to be correct, as the format was designed in the system design phase according to the user needs. For the hard copy also the output comes out as the specified requirements by the user. Hence output testing doesn't result in any correction on the system.

## **CHAPTER VII**

### **SYSTEM IMPLEMENTATION AND DEPLOYMENT**



Implementation is the process of deploying the new system in the operational environment. Proper implementation is essential to provide a reliable system to meet the organizational requirement. There are four types of implementation methods. They are Direct Changeover, Phased Implementation, Parallel Run and Pilot Approach. The most commonly used implementation methods are Pilot Approach and Parallel Run.

The system which is developed as a web application hence the other functions as normal application, as usual some web development technologies are used in the implementation of the project. The language I selected to program this software is PHP. The reason I selected PHP is that is a simple and powerful language that especially developed to create web application.

Technologies used in the development of the software are:

- ✓ Programming language – Python
- ✓ DBMS – MySQL server
- ✓ Development tool – Py charm
- ✓ Development platform – Windows 11

The front end is HTML, and CSS and back end is MySQL Server and Python. The system developed on Pycharm in Windows 10 operating system.

## **CHAPTER VIII**

### **CONCLUSION**

The “bFit-AI based fitness center” has been developed for all given conditions and it is found working effectively under the all the circumstances that may arise in the real environment. The software has been developed to reducing the operator work. This system is user friendly and is well efficient to make easy interaction with the users of system. The system is done with an insight into the necessary modifications that may be required in the future. Hence the system can be maintained successfully without much work.

## **8.1 FUTURE ENHANCEMENT**

As a future venture, it is suggested to make some changes to provide more services and to provide information at right time in right manner.

The future gym concept is based on the technologies now involved in the fitness sector. Health and fitness will remain a priority as the working environment is changing with the automation processes. The gym industry's future will focus on personalization and customization to provide each individual with a unique and tailored experience. Moreover, advanced technologies allow data tracking and analysis much more granularly. In addition, the trend towards 'boutique' gyms offering specialised services and programs is likely to continue.

The goal of this personalization will be to help people achieve their fitness goals more efficiently and effectively. In addition, they also suggest diet plans to be successful for people with similar body types and goals. All the functions have been done carefully and successfully in the software, and if any development is necessary in future, it can be done without affecting the design by adding additional modules to the system.

## **CHAPTER IX**

### **REFERENCE**

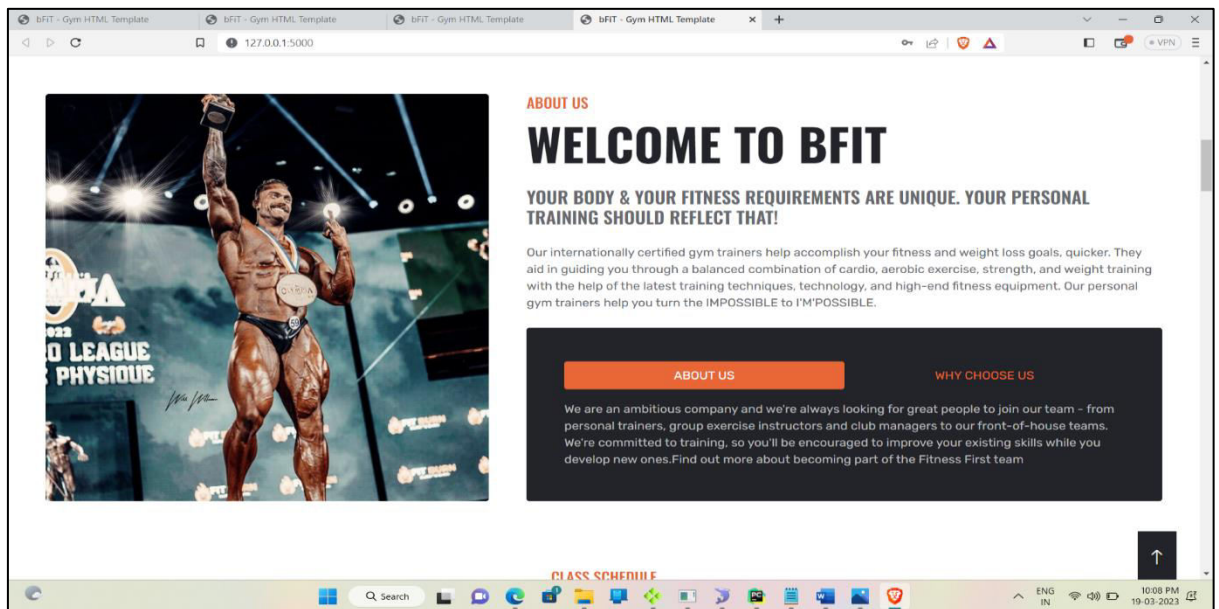
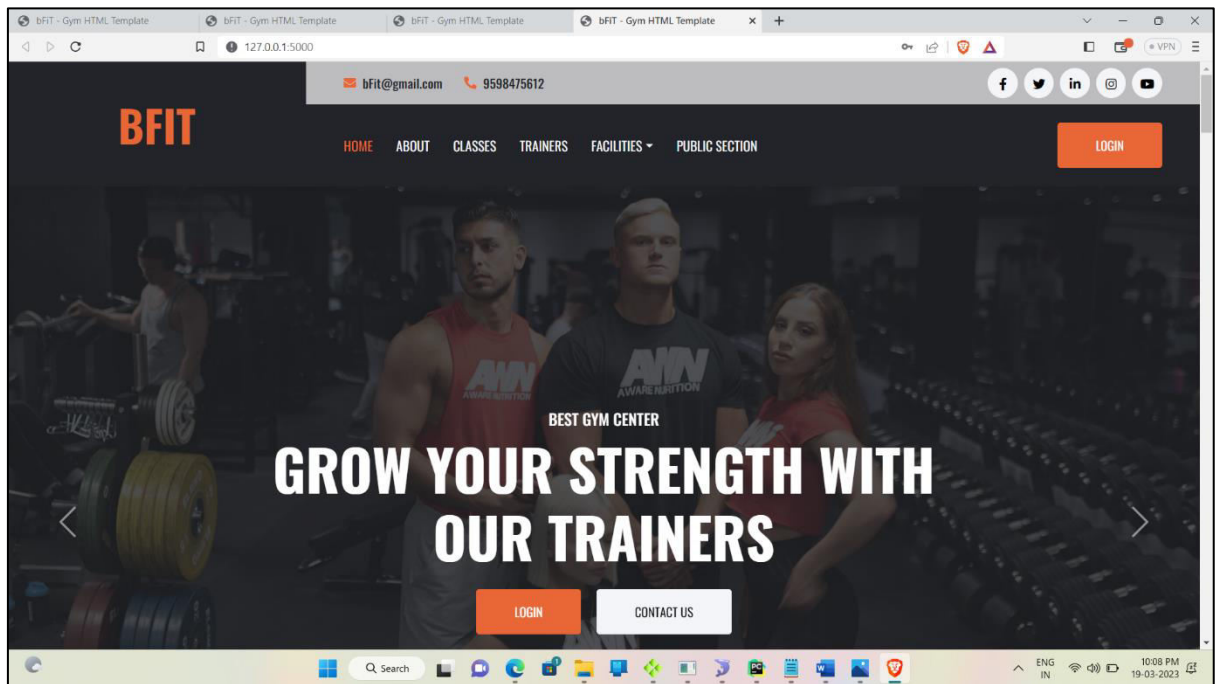
- [Google.com](https://www.google.com)
- [Youtube](https://www.youtube.com)
- [Codeacademy](https://www.codecademy.com)
- [W3school](https://www.w3schools.com)
- [Freecodecamp](https://www.freecodecamp.org)
- [code.org](https://code.org)
- [instagram](https://www.instagram.com)

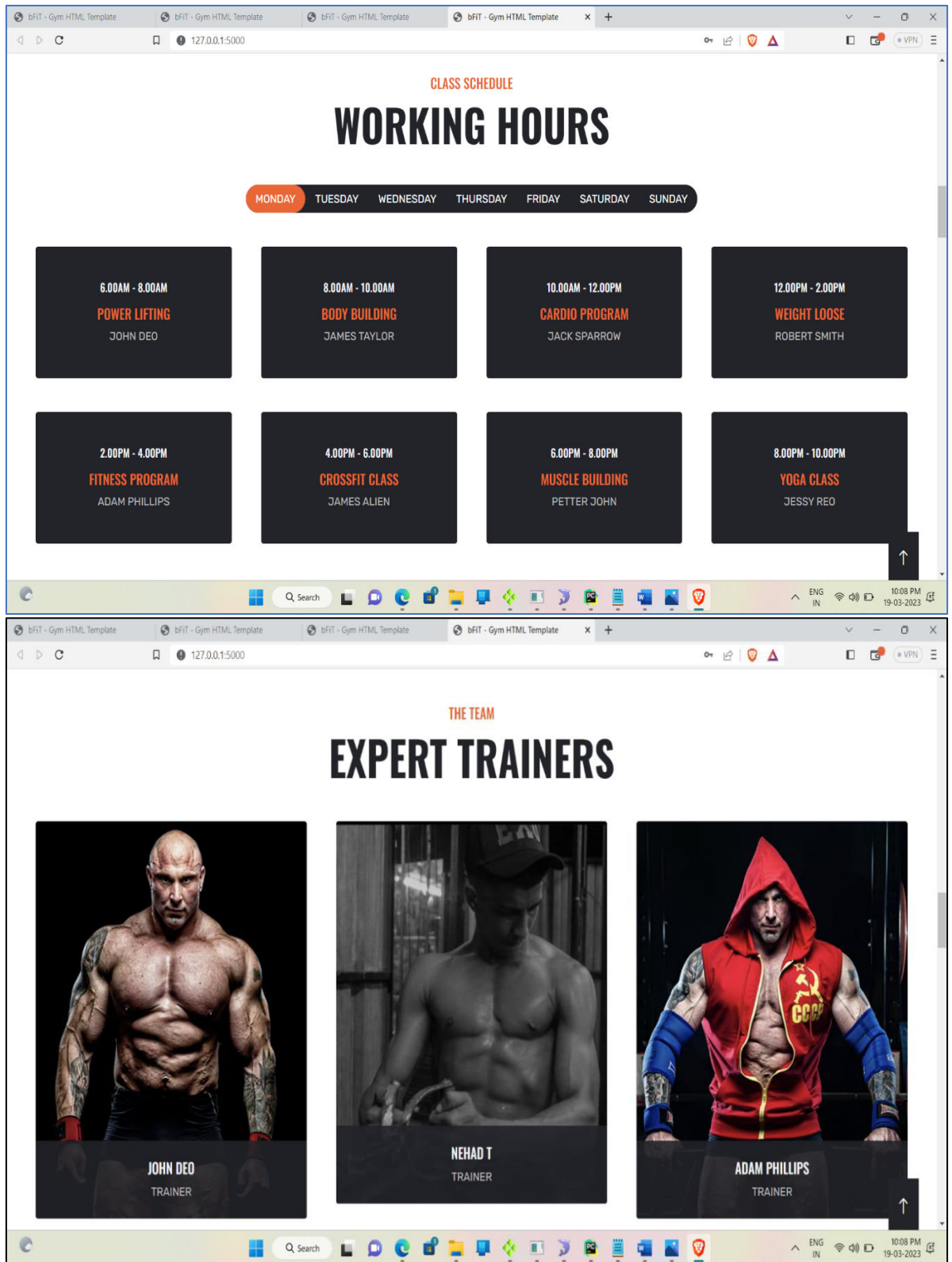
## **CHAPTER X**

### **APPENDIX**

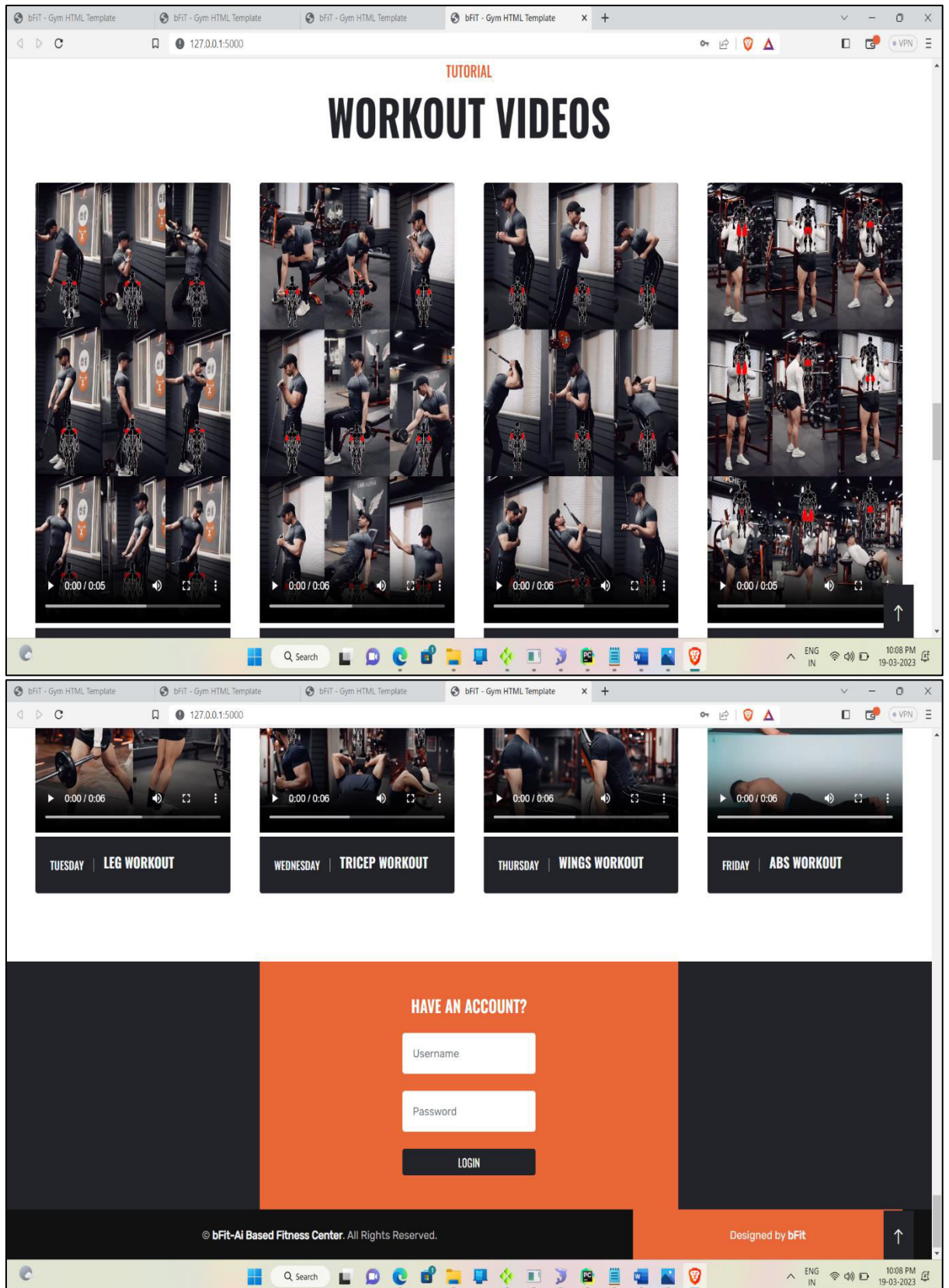
## 10.1 Screenshots

### 10.1.1 Homepage:









### 10.1.2Adminpage:

The screenshot displays the bFIT Admin page. The top navigation bar includes links for HOME, BATCH, COMPETITION, EMPLOYEE, PHYSICIAN, DOCTOR, and MORE, along with a LOGOUT button. The main header features the bFIT logo and contact information (bfit@gmail.com, 9598475612). The central banner reads "BEST GYM CENTER BUILD YOUR BODY STRONG WITH BFIT". A dropdown menu is open, showing options like Gym Requirements, View Applicants, View Attendance, View Complaints, View Feedback, View Gym Requirements, View Performer, and View User.

The main content area shows a table of users with the following data:

SL NO	NAME	EMAIL	PHONE NO	DOB	GENDER	WEIGHT	HEIGHT	BMI	PHOTOS	
1	Naveen	naveen@gmail.com	9876543210	2002-02-05	Male	50	150	22.22		ALLOCATE
2	jeswin	jeswin@gmail.com	9876543276	2002-04-28	Male	55	175	17.96		ALLOCATE
3	Arjun	arjun@gmail.com	9845672318	2002-11-19	Male	60	156	24.65		ALLOCATE
4	shanith	shanith@gmail.com	9876895432	2002-02-19	Male	58	166	21.05		ALLOCATE
5	user	user@gmail.com	8965342123	2002-11-19	Male	69	182	20.83		ALLOCATE

### 10.1.3 Instructor page:

**BFIT**

HOME BATCHES COMPETITION MEDICINE APPLICANTS PERFORMERS

LOGIN

BEST GYM CENTER

**BUILD YOUR BODY STRONG WITH BFIT**

SL NO	USER NAME	COMPETITION NAME	DATE	STATUS	
1	user	Body building	2023-05-09.	approved	
2	Naveen	WEIGHT	2023-03-02.	approved	
3	Naveen	deadlift	2023-09-01.	APPROVE	REJECT
4	Arjun	deadlift	2023-09-01.	APPROVE	REJECT
5	Arjun	Body building	2023-05-09.	APPROVE	REJECT

CLASS SCHEDULE

**WORKING HOURS**

↑

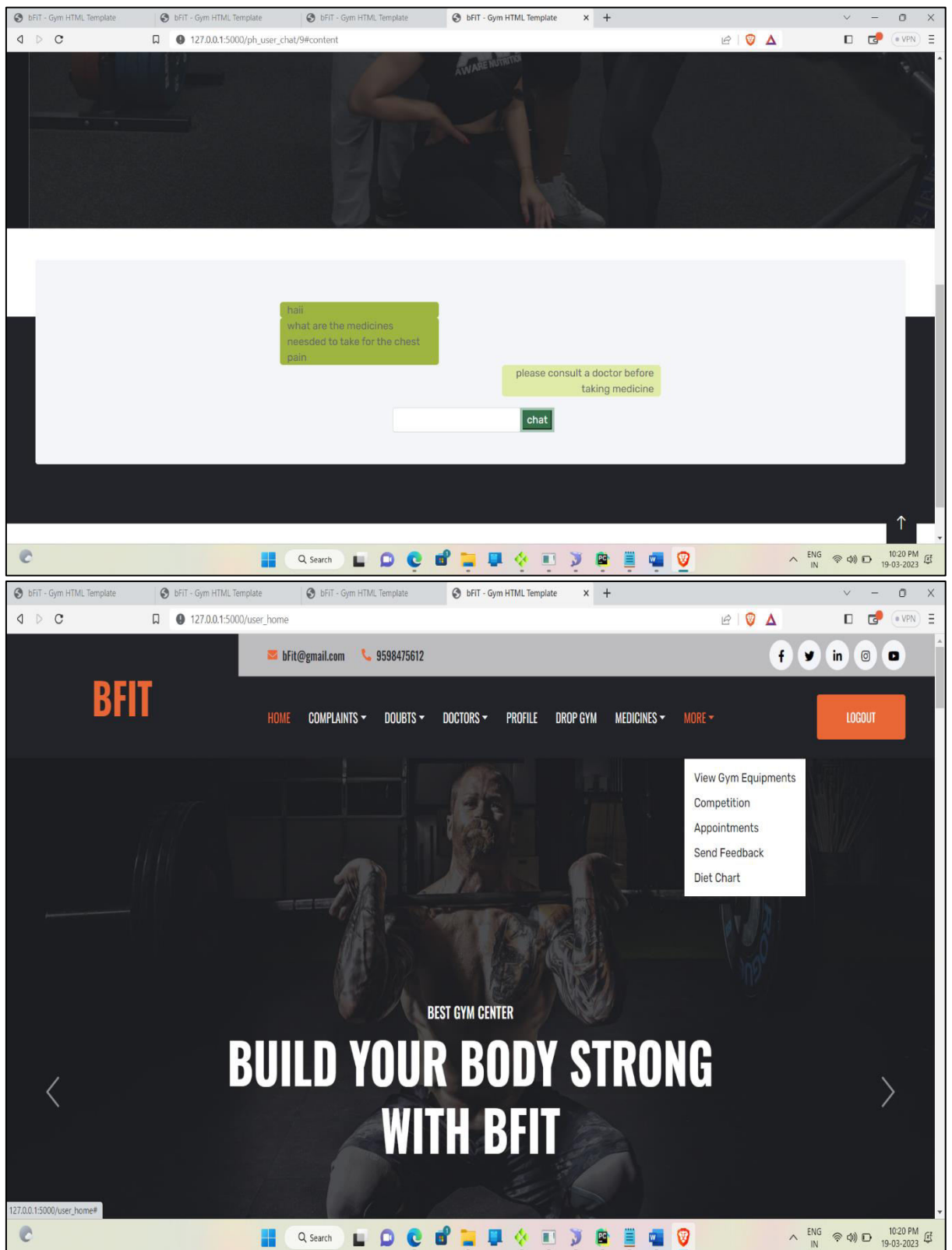
### 10.1.4 Doctor page:

The screenshot displays the bFit Doctor page. The header includes the bFit logo, navigation links (HOME, SCHEDULE, PROFILE, CHANGE PASSWORD, APPOINTMENT, REVIEW), and a LOGOUT button. The hero section features a background image of a man lifting a barbell with the text "BEST GYM CENTER" and "BUILD YOUR BODY STRONG WITH BFIT". Below the hero section is a table of appointments.

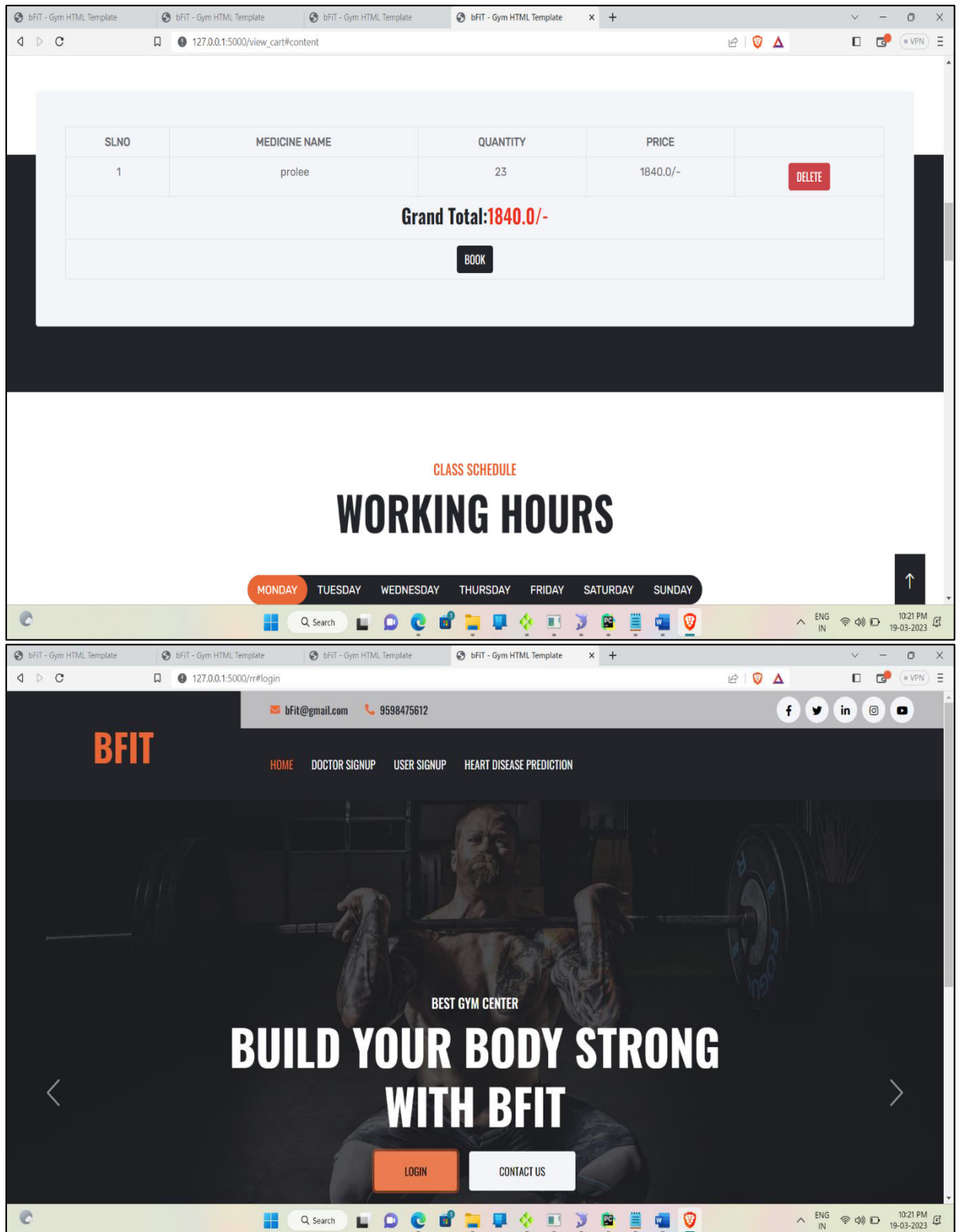
SL NO	DATE	TIME FROM	TIME TO	TOKEN NO	PATIENT NAME	PHONE	
1	2023-03-21	22:00	13:30	1	user	8965342123	CHAT
2	2023-03-21	22:00	13:30	2	Naveen	9876543210	CHAT

© bFit-AI Based Fitness Center. All Rights Reserved.

### 10.1.5 Userpage:







### 10.1.6 Heart diseasepage:

Age	<input type="text"/>
Gender	<input type="radio"/> Male <input type="radio"/> Female
Chest Pain Type	<input type="text" value="No pain"/>
Resting Blood Pressure	<input type="text"/>
Serum Cholesterol	<input type="text"/>
Fasting Blood Sugar	<input type="text" value="Sugar Level &lt;= 120mg/dl"/>
Resting Electrocardiographic Results	<input type="text" value="normal"/>
Maximum heart rate	<input type="text"/>
Exercise induced angina	<input type="text" value="Yes"/>
<input type="button" value="Submit"/>	

**You are safe now. Heart disease occurrence rate is low.**  
**Accuracy of result : 98.05**

### 10.1.7 Signup page:

The image displays two screenshots of a web application's registration interface. The top screenshot shows the 'add\_registration1#content' page, which is a user registration form. It includes input fields for Name, Email, Phone, and DOB (with a date picker), radio buttons for Gender (Male/Female), a file upload area for Photos, and input fields for Height, Weight, and Password. A green 'NEXT' button is at the bottom. The bottom screenshot shows the 'doctor\_register#content' page, which is a doctor registration form. It includes input fields for Name, Experience, Qualification, Email, Phone, and Dob (with a date picker), radio buttons for Gender (Male/Female), and a Password field. A green 'REGISTER' button is at the bottom. Both forms are presented in a clean, modern style with a light gray background and white input fields. The browser's address bar and taskbar are visible at the bottom of each screenshot.

**add\_registration1#content**

Name	<input type="text"/>
Email	<input type="text"/>
Phone	<input type="text"/>
DOB	<input type="text" value="dd-mm-yyyy"/>
Gender	<input checked="" type="radio"/> Male <input type="radio"/> Female
Photos	<input type="button" value="Choose file"/> No file chosen
Height	<input type="text"/>
Weight	<input type="text"/>
Password	<input type="password"/>
<input type="button" value="NEXT"/>	

**doctor\_register#content**

Name	<input type="text"/>
Experience	<input type="text"/>
Qualification	<input type="text"/>
Email	<input type="text"/>
Phone	<input type="text"/>
Dob	<input type="text" value="dd-mm-yyyy"/>
Gender	<input type="radio"/> Male <input type="radio"/> Female
Password	<input type="password"/>
<input type="button" value="REGISTER"/>	



**A PROJECT REPORT ON**  
**CAMPUSTALK:**  
**YOUR PERSONAL COLLEGE CHATBOT**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ATHUL THOMAS**

**REG.NO: DB20BCAR26**

**SONY MARIYA OS**

**REG.NO: DB20BCAR35**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR-670706**

**2023**

**A PROJECT REPORT ON**  
**CAMPUSTALK:**  
**YOUR PERSONAL COLLEGE CHATBOT**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**ATHUL THOMAS**

**REG.NO: DB20BCAR26**

**SONY MARIYA OS**

**REG.NO: DB20BCAR35**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR-670706**

**2023**

**DON BOSCO ARTS & SCIENCE COLLEGE**  
**ANGADIKADAVU, KANNUR**



**CERTIFICATE**

*Certified that this report titled **CAMPUSTALK: YOUR PERSONAL COLLEGE CHATBOT** is a bonafide record of the project work done by **ATHUL THOMAS (Reg.No:DB20BCAR26)** and **SONY MARIYA OS (Reg.No:DB20BCAR35)** under the supervision and guidance ,towards partial fulfilment of the requirement for award of the degree of Bachelor of Computer Application (BCA) of the Kannur university.*

Project Guide

Head of the Department

Angadikadavu

External Examiner

Date:

1.

2.

# Declaration

We ATHUL THOMAS and SONY MARIYA OS, sixth semester BCA students of Don Bosco Arts & Science College, Angadikadavu, under Kannur University do hereby declare that the project entitled “**CAMPUSTALK: YOUR PERSONAL COLLEGE CHATBOT**” is the original work carried out by us in the sixth semester under the supervision of Ms. Vineetha Mathew, lecturer of the Department of BCA, Don Bosco Arts & Science College, Angadikadavu, in partial fulfilment of the requirement for the award of the degree Bachelor of Computer Application, Kannur University.

Angadikadavu

Date

ATHUL THOMAS

SONY MARIYA OS

## **ACKNOWLEDGEMENT**

First of all we thank the lord almighty for his immense grace and blessings showered on us at every stages of this work. We are greatly indebted to our Principal Fr. Dr Francis Karackat SDB, Don Bosco Arts & Science College, Angadikadavu for providing the opportunity to take up this project as part of our curriculum.

We deeply indebted to our project guide Ms.Vineetha Mathew, lectures of department of BCA, for her assistance and valuable suggestions as guide. She made this project a reality.

We express our sincere thanks to Ms. Sindu PM, Mr. Hebin Layola, Ms. Fincy Cyriac and Ms. Sruthi N, lecturers of department of BCA, for their valuable suggestions during the course of this project. Their critical suggestions helped us to improve the project work.

Acknowledging the efforts of everyone, their chivalrous help in the course of the project preparation and their willingness to collaborate with the work, their magnanimity through lucid technical details lead to the successful completion of our project.

We would like to express our sincere thanks to all our friends, colleagues, parents and all those who have directly or indirectly assisted during this work.

# CONTENTS

<b>chapters</b>	<b>contents</b>	<b>Page no:</b>
1	Introduction	9
1.1	Model	11
2	System Analysis	15
2.1	Introduction to system analysis	16
2.2	Existing System	16
2.3	Proposed System	16
2.4	Feasibility studies	17
2.4.1	Economical feasibility	18
2.4.2	Technical feasibility	18
2.4.3	Behavioural feasibility	19
2.5	System Specifications	19
2.5.1	Software Specifications	19
2.5.2	Hardware specifications	20
2.6	Identification of Actors	21
2.7	Identification of use cases	21
2.7.1	Use case for the actor Administrator	22
2.7.2	Use case for the actor Teacher	22
2.7.3	Use case for the actor Students	23
2.8	Use case Diagrams	24
3	System Design	27
3.1	System Design- Introduction	28
3.2	Database Design	29
3.3	Table Design	30

3.4	Data Flow Diagrams	37
3.5	ER Diagrams	45
4	Coding	47
4.1	Input Interface	48
4.2	Output Interface	48
4.3	Technical Specifications	48
4.3.1	HTML	48
4.3.2	CSS	51
4.3.3	JavaScript	54
4.3.4	Android	56
4.3.5	JAVA	60
4.3.6	Python	63
4.3.7	MySQL	63
4.3.8	Pycharm	64
4.3.9	Flask	65
5	Coding pages	66
6	System Testing	83
6.1	Testing and Evaluation	84
6.2	Testing Strategies	85
6.3	Testing Techniques	86
6.3.1	White box testing	86
6.3.2	Blackbox testing	86
6.3.3	Unit testing	87
6.3.4	Integration testing	87
6.3.5	Acceptance testing	88

6.3.6	Output testing	88
7	System implementation & deployment	89
8	Conclusion	90
8.1	Future Enhancement	91
9	Reference	92
10	Appendix	93



# **1.INTRODUCTION**

Today, every organisation depends on Information and Communication Technology (ICT) for the efficient service delivery and cost-effective application of technological resources. With growing preference towards faster services and acceptance of Artificial Intelligence (AI) based tools in business operations globally as well as in India, the global Chatbot market is going to accelerate in the next decade. In the era of AI, the Chatbot market is witnessing extraordinary growth with the increased demand for smartphones and increased use of messaging applications. In the past few years, the food delivery business, finance and the Ecommerce industry have embraced Chatbot technology. One of the industries which can really benefit from using this technology is the educational sector. Education can benefit from Chatbot development. It can improve productivity, communication, learning, efficient teaching assistance, and minimize ambiguity from interaction. A new education platform can solve next-level problems in education using this technology as the engagement tool. The aim of this research paper is to find out the factors which affect the adoption of Chatbot technology in order to enhance the student learning experience in the Indian higher education sector. In this research, a Quantitative method is used through data collection from surveys of some of the prominent higher education institutes using Chatbot technology in India. It is expected that the research outcome will help Chatbot developers and higher education providers to better understand

## **OBJECTIVES**

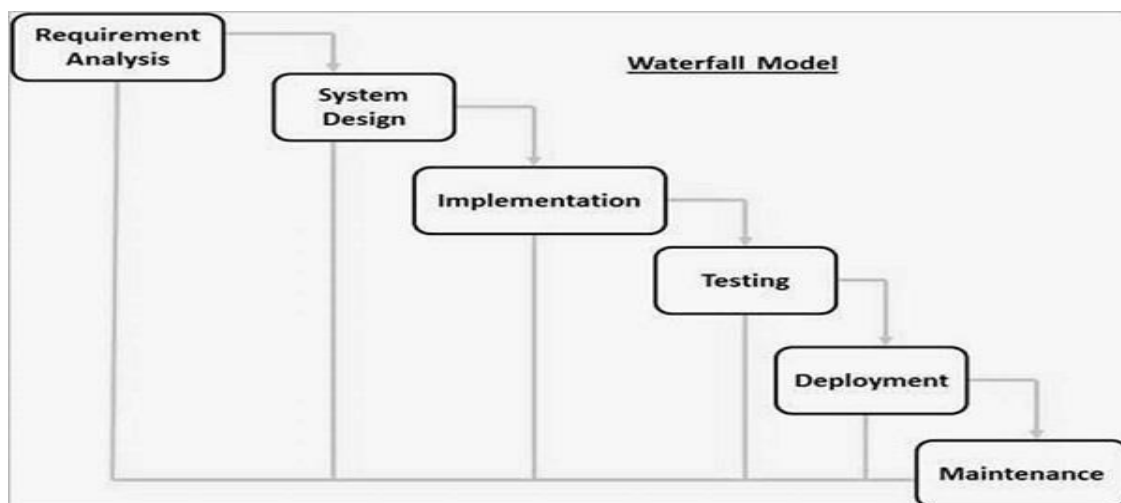
- This Program is able to simulate a conversation with the user using natural language through messaging platforms, phone applications and websites.
- Users interact with Chat bots that have a conversational user interface (CUI), which allows users to interact with the bot.

## 1.1 MODEL:

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially

### Waterfall Model - Design:

In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. Following is the pictorial representation of Iterative and Incremental model:



The sequential phases in Waterfall model are –

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

## **Waterfall Model - Application:**

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are –

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

## **The advantages of the waterfall model SDLC Model are as follows:**

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Some of the major advantages of the Waterfall Model are as follows –

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.

- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

**The disadvantages of the waterfall model SDLC Model are as follows:**

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows –

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

## **2. SYSTEM ANALYSIS**

## **2.1 INTRODUCTION TO SYSTEM ANALYSIS**

System analysis is the process of collecting and interpreting facts, understanding problems and using the information to suggest improvement on the system. This will help to understand the existing system and determine how computers make its operation more effective. The aim of this analysis is to collect detailed information on the system and the feasibility study of the proposed system.

## **2.2 EXISTING SYSTEM**

At present the higher educational institutions of India are providing information via websites. Most of the institutions have their own website with numerous data and information. But disadvantage of having such websites is information overload. It is very hard to find out the particular data we are looking for from the large websites. If we are looking at a website to figure out facts, its take too much time when compared to the chatbots. Sometimes the user is looking for a single line information ,but looking on website cause reading through paragraphs to find out that data.

Following are the major disadvantages of existing systems:

- Hard to find quality contents.
- Customer dissatisfaction.
- Users are forced to watch Ads.
- Time consuming

## **2.3 PROPOSED SYSTEM**

With chatbot technology, institutions can meet the needs and expectations of today's students. Chatbots are available 24\*7 and can provide information quickly and in many languages, making them useful at institutions across the globe. And advances in natural language



processing (NLP) have enabled computer-based applications to better interpret languages. Technological advancement has enabled organizations to conduct their daily businesses in an effective and efficient manner. The information and communication technology have been adopted to facilitate effective and timely delivery of services in different public and private sections in India including higher education. It has been established that Chatbot development can improve learning, communication, and productivity, as well as provide efficient teaching assistance and minimize ambiguity. Consequently, this study aims at establishing the factors which affect the adoption of Chatbot technology to enhance the student experience in the Indian higher education sector.

#### Advantages of proposed system over existing system:

- A truly personalised user experience.
- Re-engage with users in a relevant way
- A better and faster experience
- Balance automation with human touch
- Increase conversation rates
- Time saving to users when compared to websites

## 2.4 Feasibility Study

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioral feasibilities.

The proposed system is theoretically investigated to check the feasibility and found that they are more reliable and efficient in the cases given below. There are three aspects in the feasibility study portion of the preliminary investigation.

✓ Economic feasibility

✓ Technical feasibility

✓ Behavioural feasibility

The proposed system must be evaluated from a technical point of view first,

and if technical feasible their impact on the organization must be assessed. If compatible, the operational system can be devised. Then they must be tested for economic feasibility.

#### **2.4.1 Economic Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors which affect the development of a new system is the cost it would require. Since the system developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

#### **2.4.2 Technical Feasibility**

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, procedures, and staff. Having identified an outline system, the investigation must go on suggest the type of equipment, required method developing the system, of running the system once it has been designed. The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Though the technology become obsolete after some period, due to the fact that newer version of some software supports older versions, the system still be used. So there are only minimal constraints involved with this project. The system has been developed using C# and .NET, along with the database software SQL server, thus we could conclude that the project is technically feasible for development.

### **2.4.3 Behavioural Feasibility**

People are inherently resistant to change and computers have been known to facilitate change. The System is designed in user friendly manner and we need to provide any special training for the persons using this software. The operating system used is Windows 10, which is also user friendly. Since the application is web biased and can easily accessed in a web browser, which is quite familiar to the intended users, it does not have any operational barriers. So, no need to provide any special training for using this application software and hence it is behaviourally feasible.

## **2.5 System Specifications**

System Specification deals with the technical aspects the project has to meet in minimum to work successfully. This also includes the different aspects the software requirement is determined from. The technical details typically include:

- Software Specification
- Hardware Specification

### **2.5.1 Software Specifications**

The software required for the application depends on the following factors:

- ✓ The flexibility of the software
- ✓ Software contracts
- ✓ Limitation of the software

## **Software Requirement**

This specifies the minimum software requirements for implementing the system. This includes:

- Front end: - HTML, CSS, Python 3.9
- Back end: - SQL
- Client side: - Java Script
- Server-side scripting: - Python
- Platform: -Flask
- Operating System: -Microsoft windows 10/11

### **2.5.2 Hardware Specifications**

The software required for the application depends on the following factors:

- ✓ Determining size and capacity requirements.
- ✓ Computer evaluation and measurement.
- ✓ Financial factors.
- ✓ Maintenance and support.

## **Hardware Requirement**

- Microprocessor: Any 64-bit processor.
- Clock speed: - 2.13GHz
- Ram: 1 GB and above
- Hard disk: 40 GB and above
  
- Keyboard: - standard keyboard
- Mouse: Standard mouse
- Connectivity: - LAN & Wi-Fi

## 2.6 Identification of Actors

A use cases represents the functionality of an actor. It is defined as a set of actions performed by a system, which yields an observable result. An ellipse containing its name inside the ellipse or below it represents. it. It is placed inside the system boundary and connected to an actor with an association. This shoes how the use cases and the actor interact.

We can identify the actors through a list of questions. The answers to these questions bring out the actors of the system is.

- Admin
- Teacher
- Student

Here we need to specify the use cases of each actor.

## 2.7 Identification of use cases

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yield an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

To find out the use cases, ask the following questions to each of the actors.

- ✓ Which functions does the actor require from the system? What does the actor need to do?
- ✓ Does the actor need to read, create, destroy, modify or store some kind of information in the system?
- ✓ Does the actor have to calculate something? And want to provide information for others?
- ✓ Could the actor's daily work be simplified or made more efficient by adding new functions to the system (typically functions which are currently not automated in the system)?

### **2.7.1 Use cases for the actor Administrator**

- 1) Login:
  - i) The first step involved is login. The admin can login to the website using username and password.
- 2) Manage Teachers.
  - i) View teacher
  - ii) Block or unblock teacher
- 3) Manage students
  - i) View students
  - ii) Block or unblock students
- 4) Manage events
  - i) View events
  - ii) Add new events
- 5) Manage notification
  - i) View notifications
  - ii) Send notifications
- 6) View feedback
- 7) Manage report
  - i) View report
  - ii) Send reply

### **2.7.2 Use cases for the actor Teacher**

- 1) Login
- 2) signup
- 3) manage group
- 4) view and update profile
- 5) create and view event
- 6) share ideas
- 7) share articles
- 8) like & rate ideas and articles
- 9) discussion forum
- 10) send feedback

11) send report and view reply

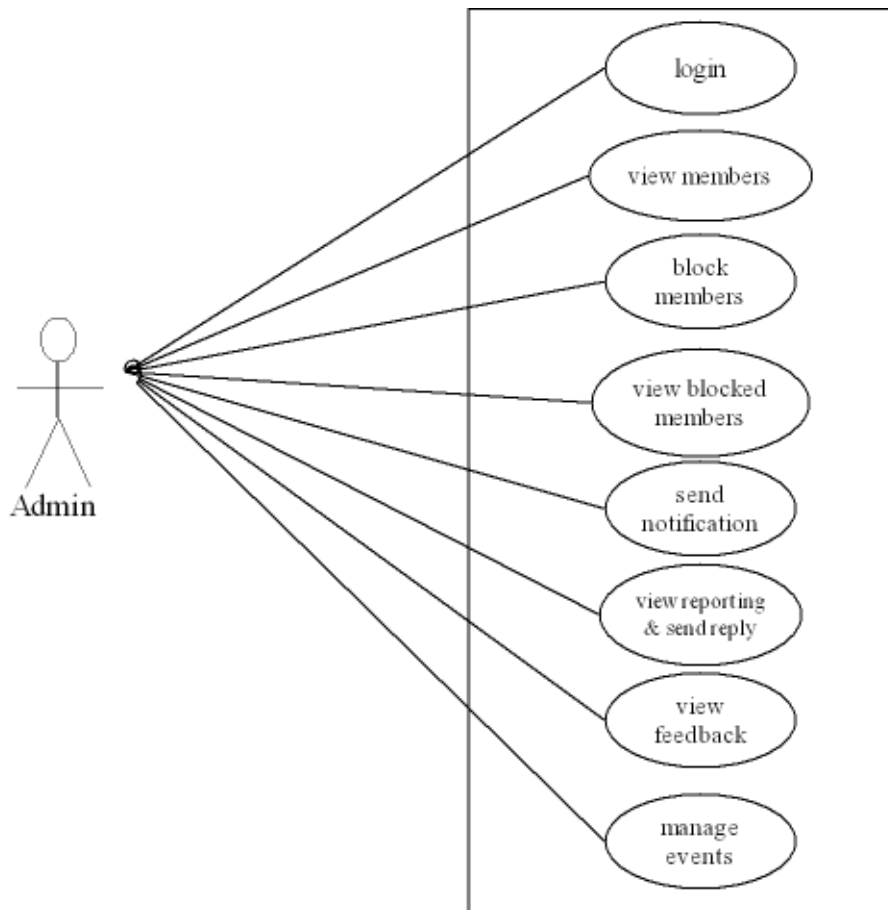
12) send notification

### **2.7.3 Use cases for the actor student**

1. Login
2. signup
3. manage group
4. view and update profile
5. view event
6. share ideas
7. share articles
8. like & rate ideas and articles
9. discussion forum
10. send feedback
11. send report and view reply
12. view notification
13. interaction with chatbot

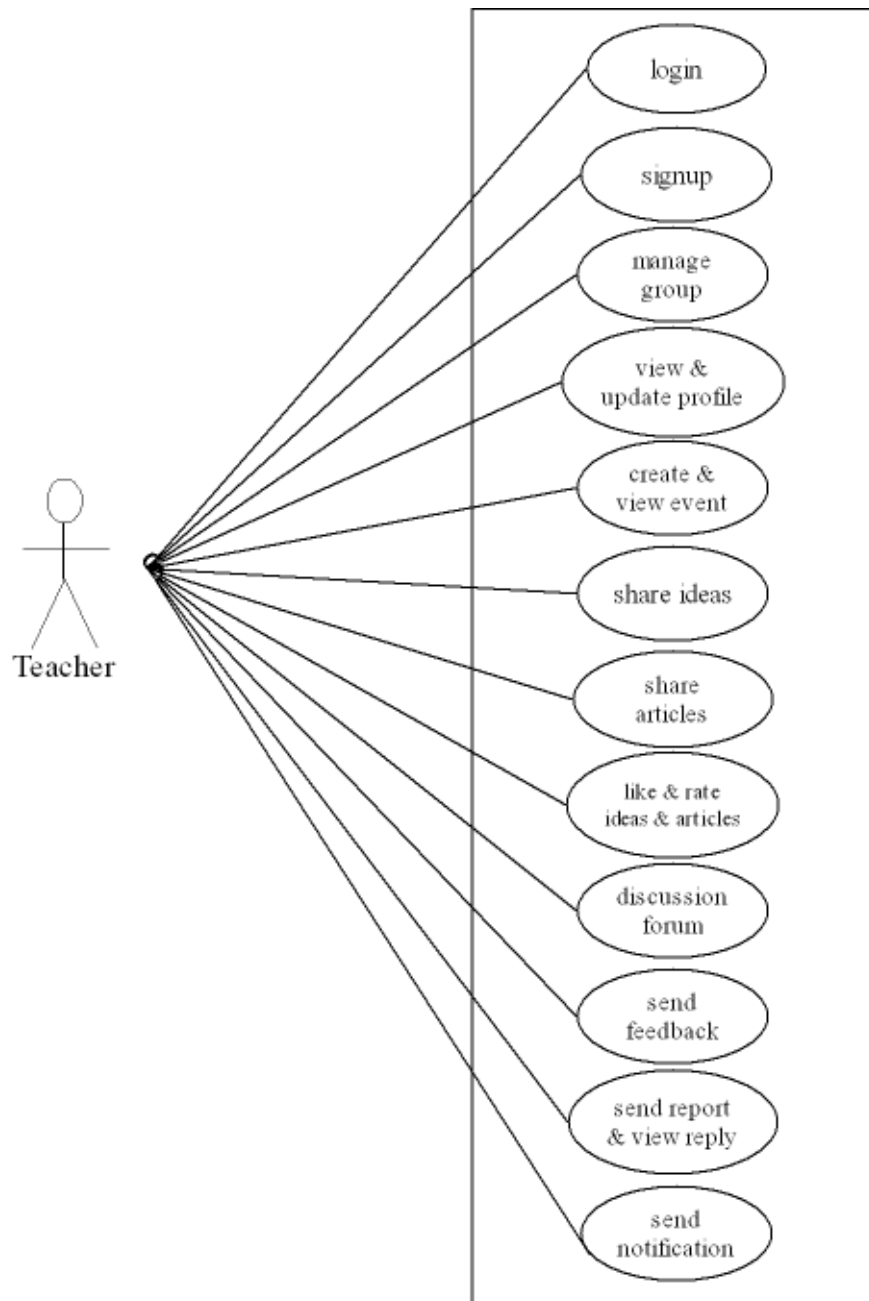
## 2.8 USE CASE DIAGRAM

### Usecase of Admin:

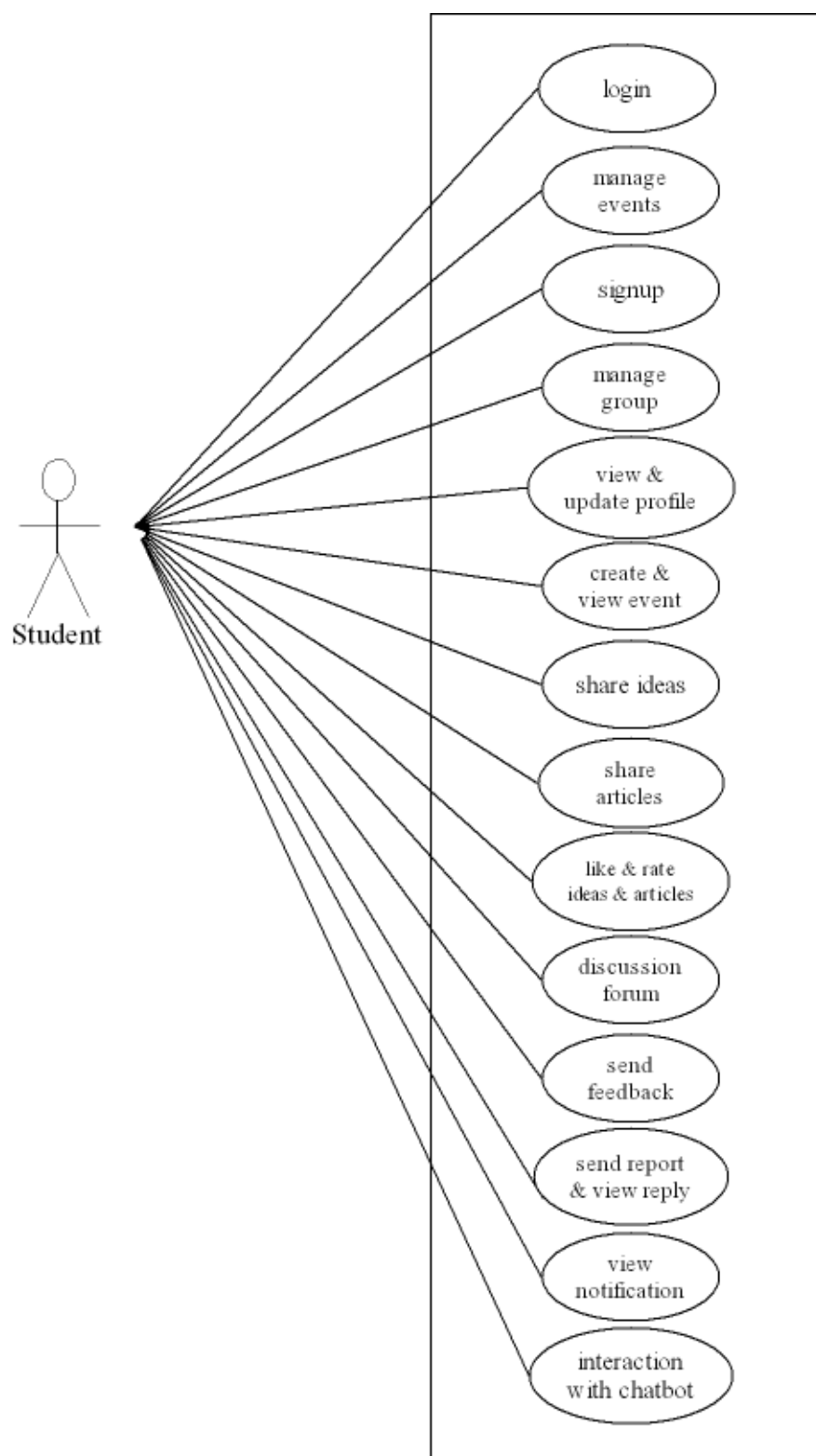




## Usecase of Teacher:



### Usecase of student:



### **3. SYSTEM DESIGN**

### **3.1 SYSTEM DESIGN - INTRODUCTION**

System Design involves translating system requirements and conceptual design into technical specification and general flow of processing. After the system requirements have been identified, information has been gathered to verify the problems and after evaluating the existing system a new system is proposed. System Design is the process of planning of new system or to replace or complement an existing system. It must be thoroughly understood about the old system determine how computers can be used to make its operations more effective.

System Design sits at technical the kernel of the system development. Once system requirements have been analysed and specified system design is the first of the technical activities – design, code generation and test that required to build and verify the software. System Design is the most creative and challenging phases of the system life cycle. The term design describes the final system and the process by which it is to be developed.

System Design is the high level strategy for solving the problem and building a solution. System Design includes decisions about the organization of the system into subsystems, the allocation of subsystems to hardware and software components and major conceptual and policy decision that forms the framework for detailed design.

### **WEBPAGE DESIGN**

In computer software development the web page design phase has an important part role. The pages are used to gather user inputs and to display the information to the user. An excellent design of pages will improve the quality of application. It will help to increase the probability of user acceptance. Efficient page design can reduce the data entry time and it helps the user to see the different controls placed in the pages. A good page design simplifies the data access and entry.

### **ANDRIOD PAGE DESIGN**

The pages are used to gather user inputs and to display the information to the user. An excellent design of pages will improve the quality of application. It will help to increase the probability of user acceptance. A good page design simplifies the data access and entry. Efficient page design can reduce the data entry time and it helps the user to see the different controls placed in the pages.

## **INPUT DESIGN**

Input Design is the process of converting the user-oriented inputs to a computer based format. The goal of designing input data is to make the automation is easy and free from errors. The design of handling input specifies how data are accepted for computer processing. Input design is art of overall system design that needs careful attention and if includes specifying the means by which actions are taken. A system user interacting through a work station must be able to tell the system whether to accept input produce a report or end processing. The major objective of the input design is to make the data entry easier, logical and error free. With this objective the screen for the system are developed. The input design requirement such user friendliness, consistent format and interactive dialogue boxes for giving the right message and help for the user at the right time are also considered for the development of the project. The data entry operator need to know the space allocated for each field, the field sequence, which must match with source document and the format in which the data is entered.

## **OUTPUT DESIGN**

A quality output is the one, which meets the requirement of the end user and presents the information clearly. In any system, the results of processing are given to the users through the outputs. In the output design it is determined how the information is to be displayed for immediate need. Output design should improve the relationship of the system with user and help in decision making. The objective of the output design is to define the format of all printed documents and of the screens that will be produced by the system. The objective of the output design is to define the format of all printed documents and of the screens that will be produced by the system. The output has been designed as per the needs of the institution.

## **3.2 DATABASE DESIGN**

A Database is a collection of inter related data stored with minimum redundancy to serve many users quickly and efficiently. In database design data independence, accuracy, privacy and security are given higher priority. Database design is an integrated approach to the file design. This activity deals with the design of the physical data base. All entities and attributes have been identified while creating the database. The database design deals with the grouping of data into number of tables so as to.

- ✓ Reduplication of data.
- ✓ Minimize storage space.
- ✓ Retrieve the data efficiently.

Following are some guidelines for the database design:

- Design a relational schema so that it is easy to explain its meaning. Do not combine attributes from multiple entry and relationship type into a single relation.
- Design the database schema so that no insertion, deletion or modification anomalies are present in the relation.
- As far as possible, avoid placing attributes in the base relation whose values may frequently be null.
- Design relation schema so that they can be joined with equality conditions on attributes that are either primary keys or foreign keys in a way that no spurious tuples are generated.

### **3.3 TABLE DESIGN**

DB design is required to manage large bodies of information. The management of data involves both the definition of the structure of storage of information and provisions of mechanism for the manipulation of information. For developing an efficient database certain conditions have to be fulfilled such as:

- Control Redundancy
- Ease of Use
- Data Independence
- Accuracy and Integrity

There are five major steps in design process:

- Identify the table and relationship.
- Identify the data that is needed for each table and relationship.
- Resolve the relationship.
- Verify the design.
- Implement the design

*1.login table*

Colum name	Data type	Constraints	Description
login_id	int	primary key	unique identifier
username	varchar(50)	not null	name of user
password	varchar(50)	not null	secret key
user_type	varchar(50)	not null	To specify the user type

*2.student table*

Column name	Data type	Constraints	Description
Student_id	Int	Primary key	Unique identifier
Name	Varchar(50)	Not null	Name of student
Place	Varchar(50)	Not null	Place of student
Image	Varchar(100)	Not null	Profile image
Pin	Int	Not null	pincode
email	Varchar(50)	Not null	Students mail id
phone	double	Not null	Phone number

### *3. Teacher table*

Column name	Data type	Constraints	Description
teacher_id	Int	Primary key	Unique identifier
Name	Varchar(50)	Not null	Name of student
Place	Varchar(50)	Not null	Place of teacher
Image	Varchar(100)	Not null	Profile image
phone	double	Not null	Phone number

### *4. Event table*

Column name	Data type	Constraints	Description
Event_id	int	Primary key	Unique identifier
name	Varchar(50)	Not null	Name of event
Date	date	Not null	Date of the event
Image	Varchar(50)	Not null	Image of the event
Details	Varchar(50)	Not null	Event details
Creator_id	int	Not null	Id of the event creator
type	Varchar(50)	Not null	Creator type



### 5. group table

Column name	Data type	Constraints	Description
Group_id	Int	Primary key	Unique identifier
Manager_id	Int	Not null	Id of creator
Type	Varchar(50)	Not null	creator type
Name	Varchar(50)	Not null	Name of group
date	date	Not null	Created date

### 6.feedback table

Column name	Data type	Constraints	Description
feedback_id	Int	Primary key	Unique identifier
sender_id	Int	Not null	Id of sender
Type	Varchar(50)	Not null	Sender type
feedback	Varchar(50)	Not null	Feedback message
date	date	Not null	Date in which feedback sends

### 7. article table

Column name	Data type	Constraints	Description
article_id	Int	Primary key	Unique identifier
sender_id	Int	Not null	Id of sender
Type	Varchar(50)	Not null	Sender type
article	Varchar(50)	Not null	Article message
date	date	Not null	Date in which article sends

8.discussion table

Column name	Data type	Constraints	Description
Discussion_id	int	Primary key	Unique identifier
Topic	Varchar(50)	Not null	Discussion topic
Date	Date	Not null	Date
From_id	int	Not null	Id of person who starts discussion

9.discussion\_sub table

Column name	Data type	Constraints	Description
Sub_id	Int	Primary key	Unique identifier
Discussion_id	Int	Not null	Id of dicussion
Sender_id	Int	Not null	Id of sender
Content	Varchar(50)	Not null	Discussion content
Date	Date	Not null	Date of discussion
status	Varchar(50)	Not null	Current status

10.notification table

Column name	Data type	Constraints	Description
Notification_id	Int	Primary key	Unique identifier
Type	Varchar(50)	Not null	Sender type
Date	Date	Not null	Date of notification
Topic	Varchar(50)	Not null	Notification topic
Notification	Varchar(50)	Not null	Content of notification
Sender_id	int	Not null	Id of sender

11.ideas table

Column name	Data type	Constraints	Description
Ideas_id	Int	Primary key	Unique identifier
Idea	Varchar(50)	Not null	Idea content
Teacher_id	Int	Not null	Id of teacher
date	date	Not null	Date of idea created

12.like table

Column name	Data type	Constraints	Description
Like_id	Int	Primary key	Unique identifier
Liked_person_id	Int	Not null	Id of liked person
Article_id	Int	Not null	Id of article
Type	Varchar(50)	Not null	Type of liked person
date	date	Not null	Date of action

13.report table

Column name	Data type	Constraints	Description
report_id	Int	Primary key	Unique identifier
type	Varchar(50)	Not null	Sender type
Member_id	Int	Not null	Sender id
Date	Date	Not null	Date of sending
Report	Varchar(50)	Not null	Report content
Reply_date	Date	Not null	Date of reply
reply	Varchar(50)	Not null	Reply content

*14.question table*

Column name	Data type	Constraints	Description
question_id	Int	Primary key	Unique identifier
question	Varchar(50)	Not null	Question content
answer	Varchar(50)	Not null	Answer content

*15.group\_member table*

Column name	Data type	Constraints	Description
Group_member_id	Int	Primary key	Unique identifier
Group_id	Int	Not null	Id of group
Stud_id	int	Not null	Id of student

*16.chat table*

Column name	Data type	Constraints	Description
Chat_id	Int	Primary key	Unique identifier
Date	Date	Not null	Date of chat
From_id	Int	Not null	Sender id
To_id	Int	Not null	Receiver id
message	Varchar(50)	Not null	Chat message

### **3.4 DATA FLOW DIAGRAM**

A graphical representation is used to describe and analyses the movement of data through a system manual or automated including the processes, Storing of data and delays in the system. Data flow diagrams are the central tool and the basis from which other components are developed.

The transformation of data, from input to output through process may be described logically and independently of the physical components associated with the system.

They are termed logical dataflow diagrams, showing the actual implementations and the movement of data between people, departments and

workstations. DFD is one of the most important modelling tools used in system design. DFD shows the flow of data through different process in the system.

#### **PURPOSE:**

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have reasonably completed model of the behavior of the system. It is important to avoid premature designs, wherein develop designs before analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

Throughout my project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In my opinion, "efficient design of the data flow and context flow diagram helps to design the system successfully without much major flaws within the scheduled time". This is the most complicated part in a project. In the designing process, my project took more than the activities in the software lifecycle. If we design a system efficiently with all the future enhancements the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as a way of expressing system requirements in graphical form. A data flow diagram also known as "bubble chare" has the purpose of clarifying system requirements and identifying major transformations

that will become programs in system design. It functionality decomposes the requirement specification down to the lowest level. Data Flow Diagram depicts the

information flow, the transformation flow and the transformations that are applied as data move from input to output. Thus DFD describes what data flows rather than how they are processed.

Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is one of the most important tools used during system analysis. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

Data Flow Diagrams are made up of a number of symbols, which represents system components. Data flow modelling method uses four kinds of symbols, which are used to represent four kinds of system components.

These are

- Process
- Data stores
- Data flows
- External entity

## **PROCESS:**

Process shows the work of the system. Each process has one or more data inputs and produce one or more data outputs. Processes are represented by rounded rectangles in Data Flow Diagram. Each process has a unique name and number. This name and number appears inside the rectangle that represents the process in a Data Flow Diagram.

## **DATA STORES:**

A data stores is a repository of data. Processes can enter data, into a store or retrieve the data from the data store. Each data has a unique name.

## DATA FLOWS

Data flows show the passage of data in the system and are represented by lines joining system components. An arrow indicates the direction of flow and the line is labelled by name of the dataflow.

## EXTERNAL ENTITIES

External entities are outside the system but they either supply input data into the system or use other systems output. They are entities on which the designer has control. They may be an organizations customer or other bodies with which the system interacts. External entities that supply data into the system are sometimes called source. External entities that use the system data are sometimes called sinks. These are represented by rectangles in the Data Flow Diagram.

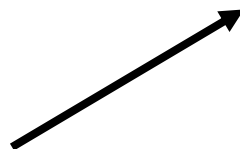
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Basic data flow diagram symbols are.....

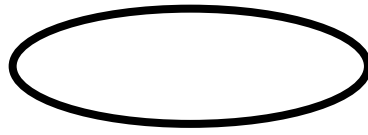
- A Square defines a source (originator) or destination of a system data:



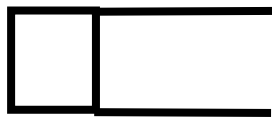
- An Arrow identifies data flow. It is a pipeline through which information flows:



- A Circle represents a process that transforms incoming data flow(s) into outgoing data flow(s):



- An Open Rectangle is a data store:



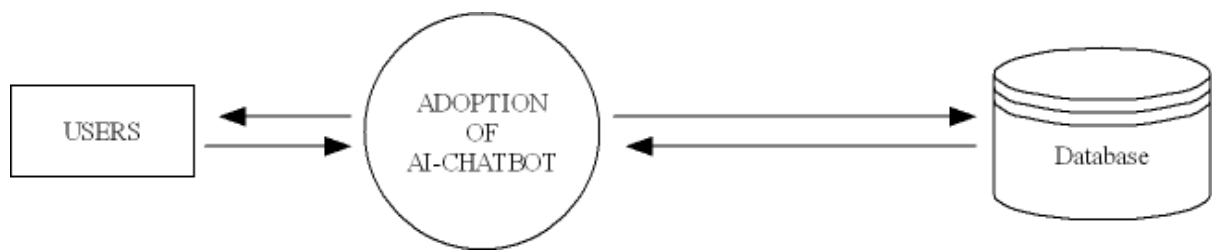
**Four steps are commonly used to construct a DFD:**

- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and left to right.
- When a process is exploded in to lower level details they are numbered.

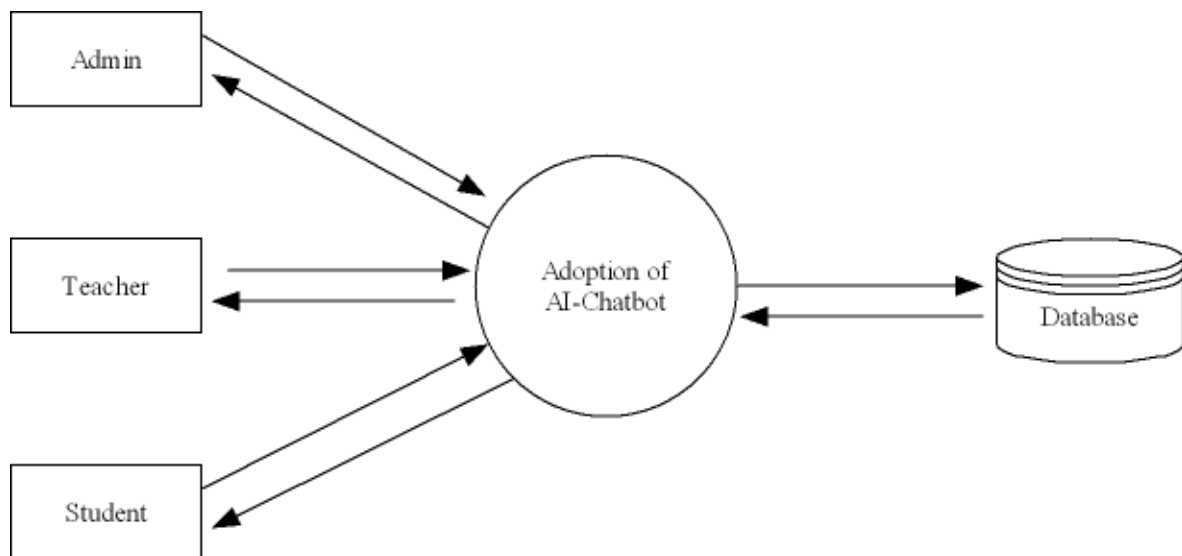
The names of data stores, sources and destinations are written in Capital letters.



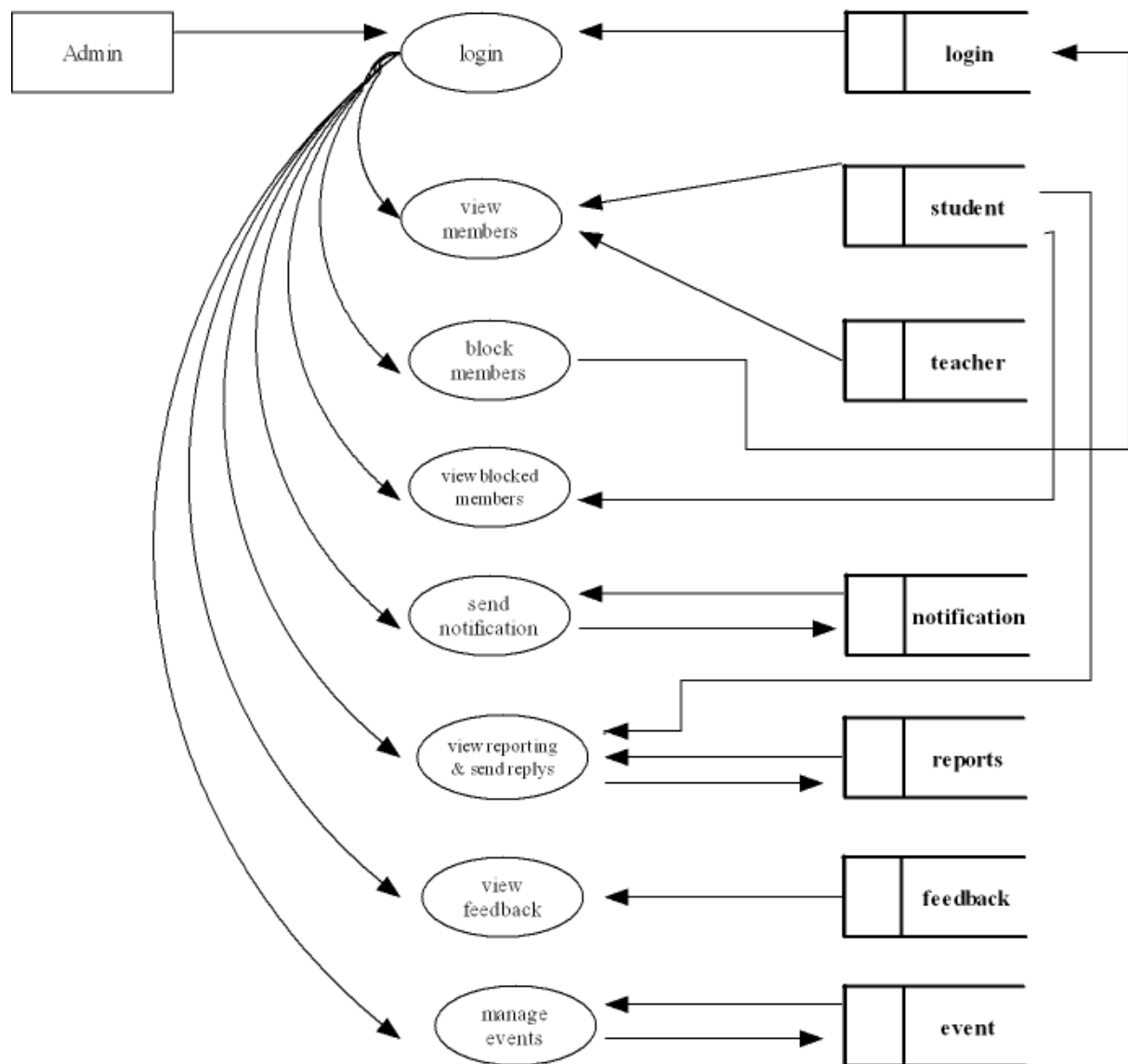
**DFD LEVEL 0:**



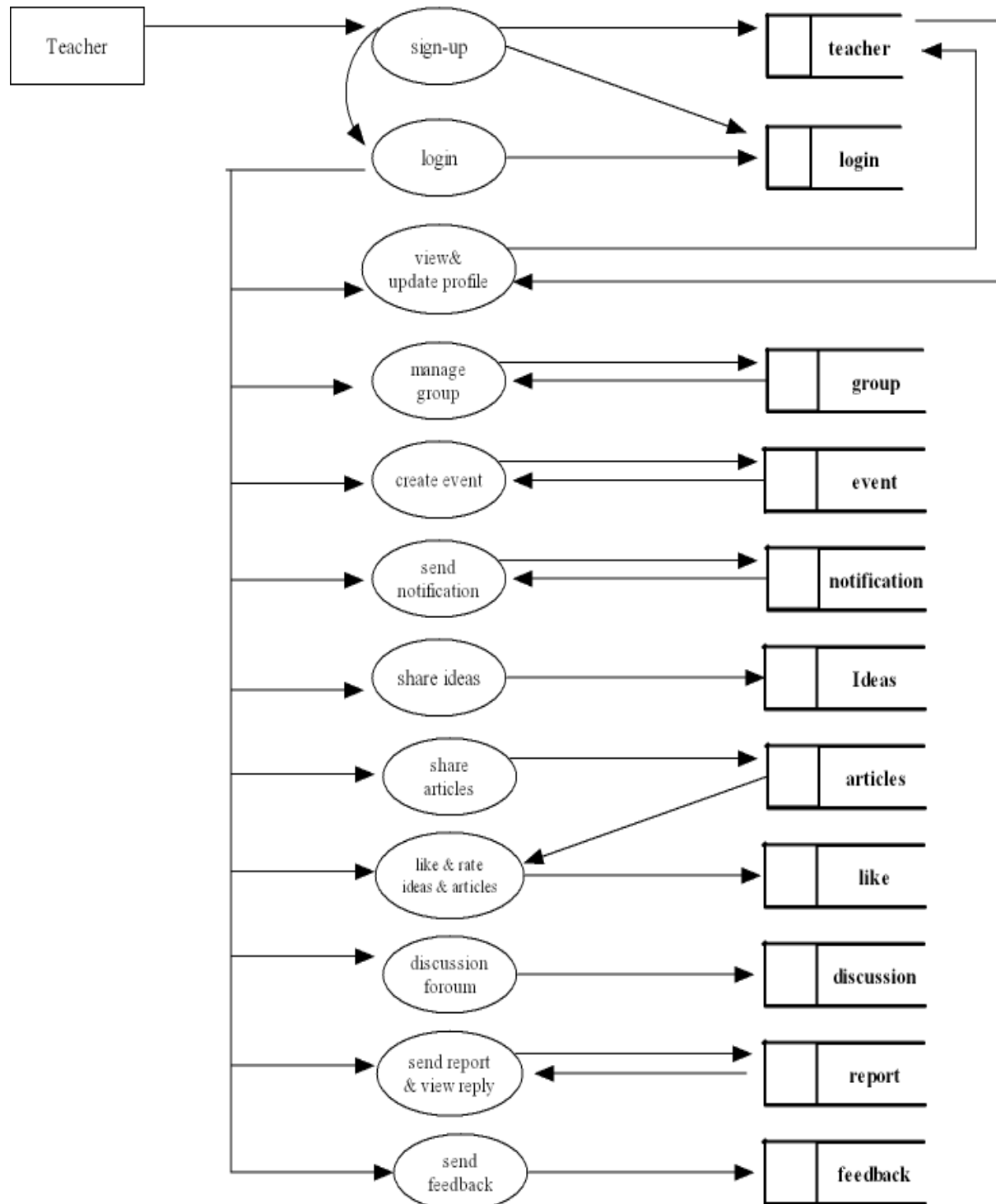
**DFD LEVEL 1:**



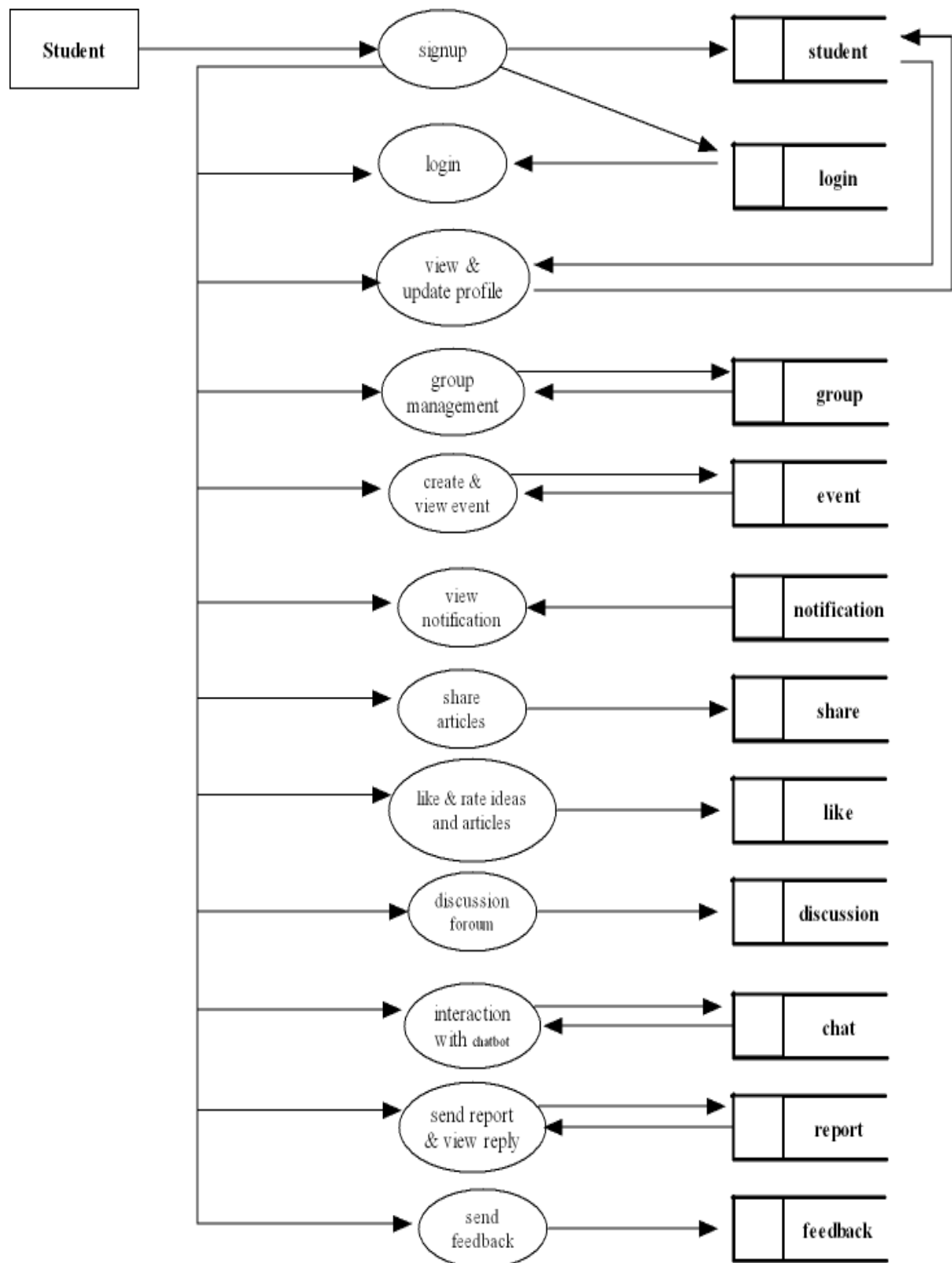
## DFD LEVEL 1.1- ADMIN



## DFD LEVEL 1.2 -TEACHER



### DFD LEVEL 1.3- STUDENT



### 3.5 ER DIAGRAM

An ER diagram is a diagram that helps to design databases in an efficient way. It is a data model for describing the data or information. It is a visual representation of data that describes how data is related to each other. The main components of ER models are entities, attributes and the relationships that can exist among them.

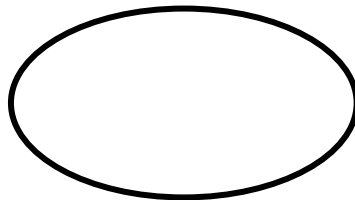
#### Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.



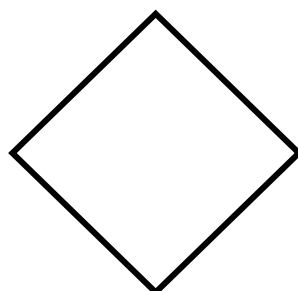
#### Attribute

Attributes are properties of entities. Attributes are represented by means of ellipses. Every ellipse represents one attribute and is directly connected to its entity (rectangle).

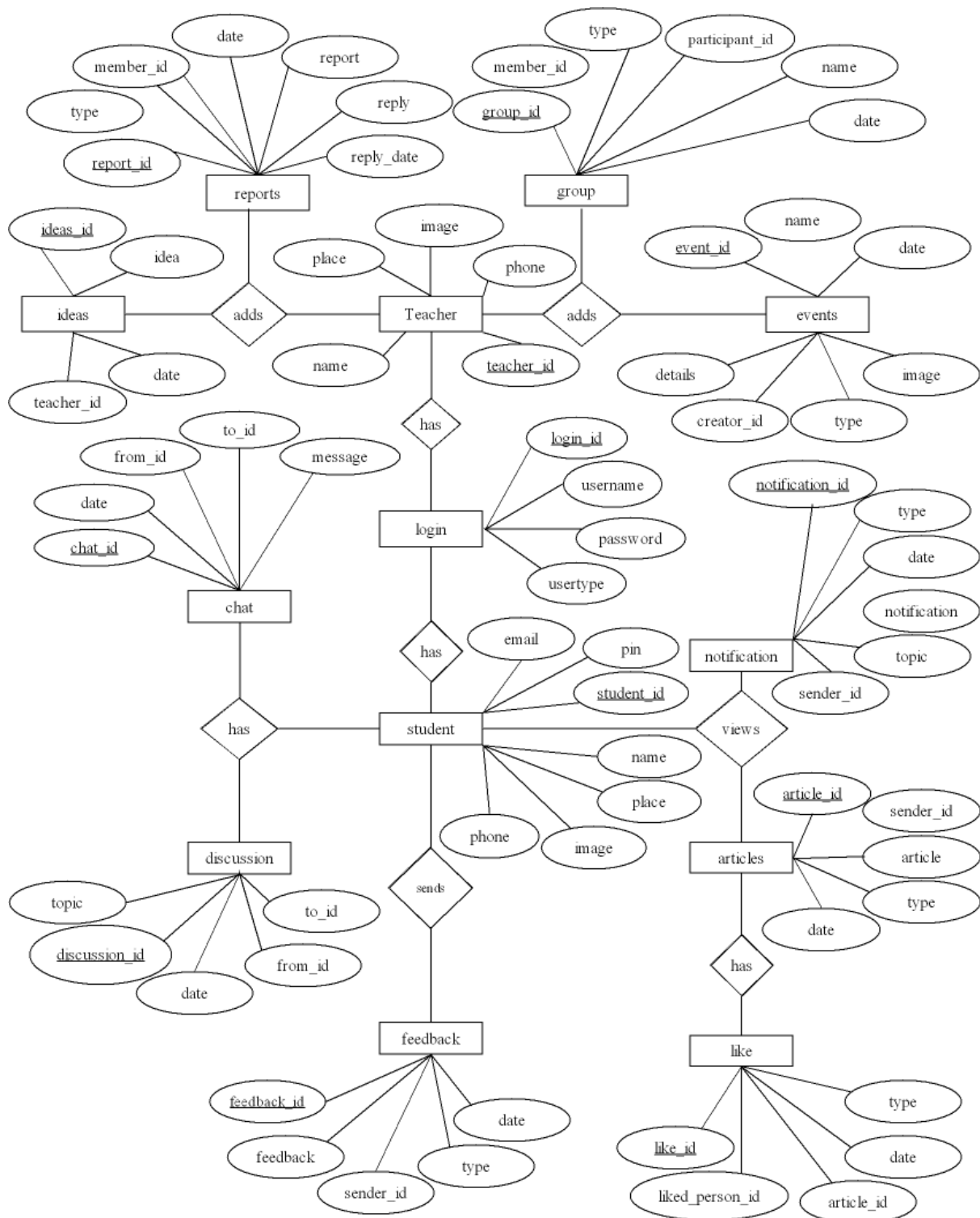


#### Relationship

Relationships are represented by diamond shaped box. Name of the relationship is written in the diamond box. All entities (rectangles), participating in relationship, are connected to it by a line.



## Architectural design



## **4.CODING**

## **4.1 INPUT INTERFACE**

Input design is a part of overall system design, which requires very careful attention. If data going into the system is correct, then the processing and output will magnify these errors. Thus the designer has a number of clear objectives in the different stages of input design.

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that input is acceptable to and understand by the user.

## **4.2 OUTPUT INTERFACE**

At the beginning of the output design various types of outputs such as external, internal, operational and interactive and turn around are defined. Then the format, content, location, frequency, volume and sequence of the outputs are specified. The content of the output must be defined in detail. The system analysis has two specific objectives at this stage.

- To interpret and communicate the results of the computer part of a system to the users in a form, which they can understand, and which meets their requirements.
- To communicate the output design specifications to programmers in a way in which it is unambiguous, comprehensive, and capable of being translated into a programming language.

## **4.3 TECHNOLOGY SPECIFICATION**

The proposed system is developed using HTML , CSS, JavaScript for web and Java(Android Studio) for Android as Front end. MySQL ,flask and python for web and Python ,MySQL for Android as Back end.

### **4.3.1 HTML**

HTML is an acronym which stands for Hyper Text Markup Language which is used for creating web pages and web applications. Let's see what is meant by Hypertext Markup Language, and Web page. Hyper Text: HyperText simply means "Text within Text." A text has a link within it, is a hypertext. Whenever you click on a link which brings you to a new webpage, you have clicked on a hypertext. HyperText is a way to link two or more web pages



(HTML documents) with each other. Markup language: A markup language is a computer language that is used to apply layout and formatting conventions to a text document. Markup language makes text more interactive and dynamic. It can turn text into images, tables, links, etc. Web Page: A web page is a document which is commonly written in HTML and translated by a web browser. A web page can be identified by entering an URL. A Web page can be of the static or dynamic type. With the help of HTML only, we can create static web pages. Hence, HTML is a markup language which is used for creating attractive web pages with the help of styling, and which looks in a nice format on a web browser. An HTML document is made of many HTML tags and each HTML tag contains different content

## **Features**

1. It is a very easy and simple language. It can be easily understood and modified.
2. It is very easy to make an effective presentation with HTML because it has a lot of formatting tags.
3. It is a markup language, so it provides a flexible way to design web pages along with the text.
4. It facilitates programmers to add a link on the web pages (by html anchor tag), so it enhances the interest of browsing of the user.
5. It is platform-independent because it can be displayed on any platform like Windows, Linux, and Macintosh, etc.
6. It facilitates the programmer to add Graphics, Videos, and Sound to the web pages which makes it more attractive and interactive.
7. HTML is a case-insensitive language, which means we can use tags either in lowercase or upper-case.

## **HTML Versions**

Since the time HTML was invented there are lots of HTML versions in market, the brief introduction about the HTML version is given below:

**HTML 1.0:** The first version of HTML was 1.0, which was the barebones version of HTML language, and it was released in 1991.

**HTML 2.0:** This was the next version which was released in 1995, and it was standard language version for website design. HTML 2.0 was able to support extra features such as form-based file upload, form elements such as text box, option button, etc.

**HTML 3.2:** HTML 3.2 version was published by W3C in early 1997. This version was capable of creating tables and providing support for extra options for form elements. It can also support a web page with complex mathematical equations. It became an official standard for any browser till January 1997. Today it is practically supported by most of the browsers.

**HTML 4.01:** HTML 4.01 version was released on December 1999, and it is a very stable version of HTML language. This version is the current official standard, and it provides added support for stylesheets (CSS) and scripting ability for various multimedia elements.

**HTML5 :** HTML5 is the newest version of Hyper Text Markup language. The first draft of this version was announced in January 2008. There are two major organizations one is W3C (World Wide Web Consortium), and another one is WHATWG( Web Hypertext Application Technology Working Group) which are involved in the development of HTML 5 version, and still, it is under development.

## **Description of HTML example**

**<!DOCTYPE>:** It defines the document type or it instruct the browser about the version of HTML

.

**<html> :** This tag informs the browser that it is an HTML document. Text between html tag describes the web document. It is a container for all other elements of HTML except<!DOCTYPE>

**<head> :** It should be the first element inside the element, which contains the metadata(information about the document). It must be closed before the body tag opens.

**<title>**: As its name suggests it is used to add title of that html page which appears at the top of the browser window. It must be placed inside the head tag and should close immediately (optional).

**<body>** : Text between body tag describes the body content of the page that is visible to the end user . This tag contains the main content of the html document.

**<h1>** : Text between <h1> tag describes the first level heading of the webpage.

**<p>**: Text between <p> tag describes the paragraph of the webpage.

### 4.3.2 CSS (Cascading Style Sheet)

CSS is used to control the style of a web document in a simple and easy way. CSS is the acronym for "**Cascading Style Sheet**". **Cascading Style Sheets (CSS)** is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript .

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

### History of CSS

CSS was first proposed by **HakonWium Lie** on October 10, 1994. At the time, Lie was working with Tim Berners-Lee (father of Html) at CERN. The European Organization for Nuclear Research is known as CERN. Hakonwium lie is known as father of css. CSS was proposed in 1994 as a web styling language, to solve some of the problems of Html 4. There

were other styling languages proposed at this time, such as Style Sheets for Html and JSSS but CSS won.

## Why to learn CSS?

**Cascading Style Sheets**, fondly referred to as **CSS**, is a simple design language intended to simplify the process of making web pages presentable.<sup>23</sup> CSS is a **MUST** for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain.

I will list down some of the key advantages of learning CSS:

- **Create Stunning Web site** - CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.
- **Become a web designer** - If you want to start a carrier as a professional web designer, HTML and CSS designing is a must skill.
- **Control web** - CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.
- **Learn other languages** - Once you understand the basic of HTML and CSS then other related technologies like javascript, php, or angular are become easier to understand.

## Types of CSS

There are three ways of inserting a style sheet in any Html documents, they are given below:

- Inline style sheet

- Internal style sheet
- External style sheet

Inline CSS is use with any elements of HTML where it is used on page. Here we use inline CSS for paragraph, the example shows how to change the color and the left margin of a paragraph. An internal style sheet should be used when a single document has a unique style. Internal styles sheet is defined in the head section of an HTML page, by using the <style> tag. An external style sheet is ideal when the style is applied to many pages. With an external style sheet, we can change the look of an entire Web site by changing one file.

## **CSS Selectors**

Selectors are used for select an Html element it is select by name, id, class etc.

1. id selector
2. class selector
3. Element Selector
4. Group Selector
5. Universal Selector

## **Features**

1. A style rule consists of a selector component and a declaration block component.
2. The selector is used to point to the HTML component which you want to get styled.
3. Inside the declaration block, one or more declarations are contained along with semicolons.
4. Every declaration which is put has a CSS property name, a semicolon, and a value. For example, color is the property and the value is red in color. Font size is the property and the 15px is the value.
5. CSS declaration ends with a semicolon and these blocks are surrounded by curly braces.

6. CSS selectors are the ones which are used to find HTML elements which are based on the element name, id, attribute, class and more.
7. One unique element will be selected by the ID of an element.
8. If you wish to select the particular element with a specific id, the # function along with the id attribute should be used.
9. If you wish to select the elements with a specific class, the period character along with the name class should be written.
10. Universal selector: If you are not interested in choosing the elements of a certain type, the universal selector simply matches with the element name.
11. Element selector: These selectors choose the element based on the element name.
12. Descendent selector: When a particular element lies inside another element, then it is called as the descendent selector.
13. ID selector: This selector uses the id of the HTML element so that a specific element could be selected.
14. Class selectors: It selects the element with a specific class attribute.
15. Grouping selectors: It will be a good option to group the selectors so as to minimize the code. Each selector along with a comma should be used to group the selectors.

### 4.3.3 JAVASCRIPT

**JavaScript** is a object-based scripting language and it is light weighted. It is first implemented by Netscape (with help from Sun Microsystems). JavaScript was created by Brendan Eich at Netscape in 1995 for the purpose of allowing code in web-pages (performing logical operation on client side).It is not compiled but translated. JavaScript Translator is responsible to translate the JavaScript code which is embedded in browser.Netscape first introduced a JavaScript interpreter in Navigator 2. The interpreter was 25 an extra software component in the browser that was capable of interpreting JavaScript source code inside an

HTML document. This means that web page developers no need other software other than a text editor of develop any web page.

## **Why we use JavaScript?**

Using HTML we can only design a web page but you cannot run any logic on web browser like addition of two numbers, check any condition, looping statements (for, while), decision making statement (if-else) at client side. All these are not possible using HTML so for perform all these task at client side you need to use JavaScript

## **History**

JavaScript is an object-based scripting language and it is light weighted. It is first implemented by Netscape (with help from Sun Microsystems). JavaScript was created by Brendan Eich at Netscape in 1995 for the purpose of allowing code in web-pages (performing logical operation on client side). Using HTML we can only design a web page but you cannot run any logic on web browser like addition of two numbers, check any condition, looping statements (for, while), decision making statement (if-else) etc. All these are not possible using HTML so to perform all these task, we use JavaScript. Using HTML we can only design a web page if we want to run any programs like c programming we use JavaScript. Suppose we want to print sum of two numbers then we use JavaScript for coding.

## **Features**

- JavaScript is an object-based scripting language.
- Giving the user more control over the browser.
- It Handling dates and time.
- It Detecting the user's browser and OS
- It is light weighted.
- JavaScript is a scripting language and it is not java.

- JavaScript is interpreter based scripting language.
- JavaScript is case sensitive.
- JavaScript is object based language as it provides predefined objects.
- Every statement in JavaScript must be terminated with semicolon (;).
- Most of the JavaScript control statements syntax is same as syntax of control statements in C language. An important part of JavaScript is the ability to create new functions within scripts. Declare a function in JavaScript using function keyword

#### 4.3.4 ANDROID

**Android** is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language. Android applications are written in the Java programming language. The Android SDK tools compile the code—along with any data and resource files— into an Android package, an archive file with an .apk suffix. All the code in a single .apk file is considered to be one application and is the file that Android-powered devices use to install the application. Application components are the essential building blocks of an Android application. Each component is a different point through which the system can enter your application. Not all components are actual entry points for the user and some depend on each other, but each one exists as its own entity and plays a specific role—each one is a unique building block that helps define application's overall behavior.

#### Features

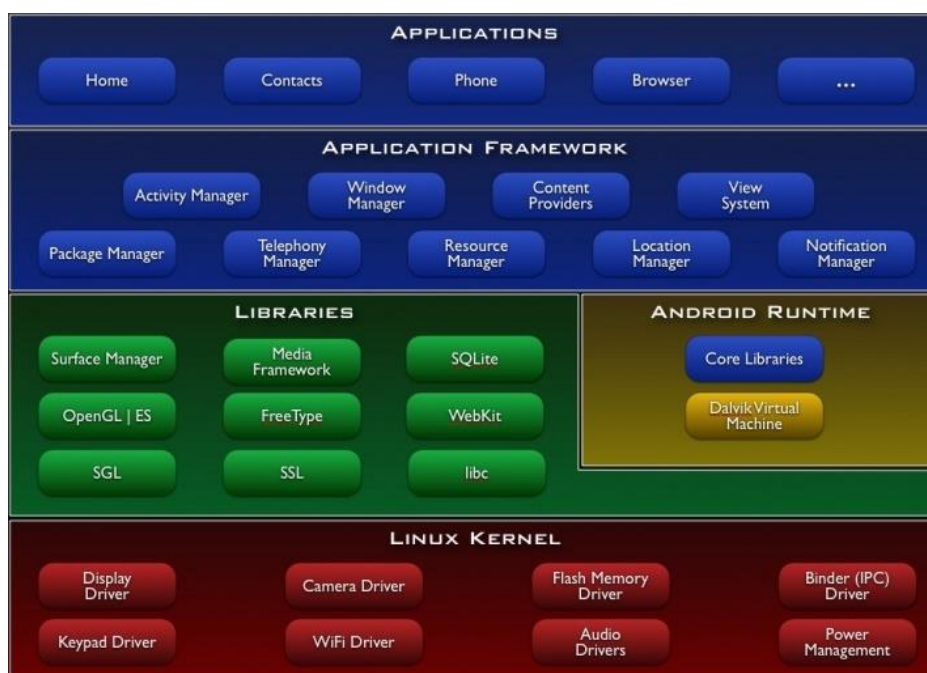
- Application framework enabling reuse and replacement of components
- Dalvik virtual machine optimized for mobile devices
- Integrated browser based on the open source Web Kit engine



- Optimized graphics powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
- Media support for common audio, video, and still image formats (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
- GSM Telephony (hardware dependent)
- Bluetooth, EDGE, 3G, and Wi-Fi (hardware dependent)
- Camera, GPS, compass, and accelerometer (hardware dependent)
- Rich development environment including a device emulator, tools for debugging, memory and performance profiling, and a plug-in for the Eclipse IDE.

## ANDROID ARCHITECTURE

The following diagram shows the major components of the Android operating system. Each section is described in more detail below.



## **APPLICATION FRAMEWORK**

By providing an open development platform, Android offers developers the ability to build extremely rich and innovative applications. Developers are free to take advantage of the device hardware, access location information, run background services, set alarms, add notifications to the status bar, and much, much more. Developers have full access to the same framework APIs used by the core applications. The application architecture is designed to simplify the reuse of components; any application can publish its capabilities and any other application may then make use of those capabilities (subject to security constraints enforced by the framework). This same mechanism allows components to be replaced by the user.

Underlying all applications is a set of services and systems, including:

A rich and extensible set of the views that can be used to build an application, including lists, grids, text boxes, buttons, and even an embeddable web browser

Content Providers that enable applications to access data from other applications (such as Contacts), or to share their own data

A Resource Manager, providing access to non-code resources such as localized strings, graphics, and layout files

A Notification Manager that enables all applications to display custom alerts in the status bar

An Activity Manager that manages the lifecycle of applications and provides a common navigation back stack.

## **Libraries**

Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below:

- System C library - a BSD-derived implementation of the standard C system library (libc), tuned for embedded Linux-based devices
- Media Libraries - based on Packet Video's Open CORE; the libraries support playback and recording of many popular audio and video formats, as well as

static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG

- Surface Manager - manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications
- LibWebCore - a modern web browser engine which powers both the Android browser and an embeddable web view
- SGL - the underlying 2D graphics engine
- 3D libraries - an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer
- Free Type - bitmap and vector font rendering<sup>24</sup>

## **Android Runtime**

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language. Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple VMs efficiently. The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been transformed into the .dex format by the included "dx" tool. The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management. Linux Kernel Android relies on Linux version 2.6 for core system services such as security, memory management, process management, network stack, and driver model. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack.

## **Activity Lifecycle**

Activities in the system are managed as an activity stack. When a new activity is started, it is placed on the top of the stack and becomes the running activity -- the previous activity always remains below it in the stack, and will not come to the foreground again until the new activity exits.

An activity has essentially four states:

- If an activity in the foreground of the screen (at the top of the stack), it is active or running.
- If an activity has lost focus but is still visible (that is, a new non-full sized or transparent activity has focus on top of your activity), it is paused. A paused activity is completely alive (it maintains all state and member information and remains attached to the window manager), but can be killed by the system in extreme low memory situations.
- If an activity is completely obscured by another activity, it is stopped. It still retains all state and member information, however, it is no longer visible to the user so its window is hidden and it will often be killed by the system when memory is needed elsewhere.
- If an activity is paused or stopped, the system can drop the activity from memory by either asking it to finish, or simply killing its process. When it is displayed again to the user, it must be completely restarted and restored to its previous state.

## **Android Studio**

**Android Studio** is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (ADT) as primary IDE for native Android application development. Android Studio supports all the same programming languages of IntelliJ, and PyCharm and Android Studio 3.0 supports Java 7 language features and a subset of Java 8 language features that vary by platform version. Features like Gradle-based build support, Android-specific refactoring and quick fixes, a rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations, Android Virtual Device (Emulator) to run and debug apps in the Android studio, etc. are provided in the current stable version.

### **4.3.5 JAVA**

The first version of Java began in 1991 and was written in 18 months at Sun micro system. In fact, it wasn't even called Java in those days it was Oak, and it was used internally at sun.

Java had adopted a model that made it perfect for the Internet, the byte code model. It is implemented as the Java virtual Machine (JVM), which is the application that actually runs the java program. 26 When JVM is installed on a computer, it can run java programs. Java programs, before ,don't need to be self-sufficient, and they don't have to include all the machine –level code that actually runs on the computer .In this way ,our Java program can be very small, because all the machine-level code to run our program is already on the target computer and doesn't have to be downloaded.

When it executes a program, the JVM can strictly monitor what goes on, which makes it great for Internet Applications.

## **JAVA FEATURES**

The inventors of java wanted to design a language, which could offer solutions to some of the problems encountered in modern programming. They wanted the language to be reliable, portable and distributed but also simple, compact and interactive. Sun Microsystems officially describes java with the following attributes.

- Compiled and Interpreted
- Platform-Independent and Portable
- Object-Oriented
- Robust and Secure
- Distributed
- Familiar, Simple and Small
- Multithreaded and Interactive
- High Performance
- Dynamic and Extensible

## **Compiled and Interpreted**

Usually a computer language is either compiled or interpreted. Java combines both these approaches thus making java a two-stage system first java compiler translate source code in to byte code instructions. Byte-codes are not machine code that can be directly executed by the machine that is running the java program. Platform Independent and Portable.

The most significant contribution java over other languages is its portability. Java programs can be easily moved from one computer system to another, anywhere at any time. Changes and Upgrades in 27 operating systems, processors and system resources will not force any changes in java programs. This is the reason why java has become a popular language for programming on internet, which interconnects different kinds of systems worldwide. Java ensures portability in two ways. First, java compiler generates byte code instructions that can be implemented on any machine. Secondly, the sizes of the primitive data types are machine independent.

## **Object-Oriented**

Java is a true Object-Oriented Language. Almost everything in Java is an Object. All program code and data reside within objects and classes. Java comes with an extensive set of classes arranged in packages that we can use in our programs by inheritance. The object model in java is simple and easy to extend.

## **Robust and Secure**

Java is a robust language. It provides many safe guards to ensure reliable code. It has strict compile time checking for data types. It is designed as garbage collected language. Java also incorporates with the concept of exception handling.

## **Distributed**

Java is designed as a distributed language for creating application on network. It has the ability to share both data & Program.

## **Multithreaded and interactive**

Multithreaded means handling multiple tasks simultaneously java supports multithreaded programs. This means that we not wait for the application to finish one task before beginning another.

## **High performance**

Java performance is impressive for an interpreted language mainly due to the use of intermediate byte code. Java architecture is also designed to reduce overheads during runtime.

### 4.3.6 PYTHON

Python is an object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

### 4.3.7 MySQL

MySQL software is Open Source

Open source means that it is possible for anyone to use and modify. Anybody can download the MySQL software from the Internet and use it without paying anything. The MySQL database server is very fast, reliable, and easy to use. It was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Though under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet

- An object-oriented interface

- Support for prepared statements
- Support for multiple statements
- Support for transactions
- Enhanced debugging support
- Embedded server support

### 4.3.8 PYCHARM:

PyCharm is an [integrated development environment](#) (IDE) used in [computer programming](#), specifically for the [Python](#) language. It is developed by the [Czech](#) company [JetBrains](#) (formerly known as IntelliJ).<sup>[5]</sup> It provides code analysis, a graphical debugger, an integrated unit tester, integration with [version control systems](#) (VCSes), and supports web development with [Django](#) as well as [data science](#) with [Anaconda](#).<sup>[6]</sup>

### FEATURES:

- Coding assistance and [analysis](#), with [code completion](#), syntax and error highlighting, [linter integration](#), and quick fixes
- Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
- Python [refactoring](#): includes rename, extract method, introduce variable, introduce constant, pull up, push down and others
- Integrated Python [debugger](#)
- Integrated [unit testing](#), with line-by-line [code coverage](#)
- Version control integration: unified user interface for [Mercurial](#), [Git](#), [Subversion](#), [Perforce](#) and [CVS](#) with change lists and merge



### 4.3.9 FLASK

**Flask** is a micro framework written in python. It is classified as [microframework](#) because it does not require particular tools or libraries.<sup>[2]</sup> It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

## **5.CODING PAGES**

## Chatbot.py

```
import base64
import random

from flask import Flask, render_template, request, redirect, jsonify
from DBConnection import Db
import datetime

app = Flask(__name__)
static_path=r"C:\Users\Athul Thomas\PycharmProjects\chatbot\static\"

@app.route('/')
def hello_world():
    return render_template("login_index.html")

@app.route("/login_post", methods=['post'])
def login_post():
    username=request.form['textfield']
    password=request.form['textfield2']
    db=Db()
    res=db.selectOne("select * from login where username='"+username+"' and
password='"+password+"'")
    if res is not None:
        if res['usertype']=="admin":
            return redirect("/admin_home")
        else:
            return "invalid"
    else:
        return "invalid"

@app.route("/admin_home")
def admin_home():
    return render_template("admin/admin_index1.html")
    # return render_template("admin/home.html")

@app.route("/admin_add_question")
def admin_add_question():
    return render_template("admin/Add_questions.html")

@app.route("/admin_add_question_post", methods=['post'])
def admin_add_question_post():
    qn=request.form['textfield']
```

```

    ans=request.form['textfield2']
    db=Db()
    db.insert("insert into question(question, answer) values('"+qn+"','"+ans+"')")
    return '<script>alert("added
successfully");window.location="/admin_add_question"</script>'

@app.route("/admin_view_question")
def admin_view_question():
    db=Db()
    res=db.select("select * from question")
    return render_template("admin/view_question.html", data=res)

@app.route("/admin_delete_question/<qid>")
def admin_delete_question(qid):
    db=Db()
    db.delete("delete from question where question_id='"+qid+"'")
    return redirect("/admin_view_question")

@app.route("/admin_add_event")
def admin_add_event():
    return render_template("admin/Add_event.html")

@app.route("/admin_add_event_post",methods=['post'])
def admin_add_event_post():
    name=request.form['textfield']
    date=request.form['textfield2']
    image=request.files['fileField']
    details=request.form['textarea']
    db=Db()
    date=datetime.datetime.now().strftime("%d%m%y-%H%M%S")
    image.save(static_path + "pic\\"+date+'.jpg')
    path="/static/pic/"+date+'.jpg'
    db.insert("insert into event(name, date,image,details,creator_id,type)
values('"+name+"','"+date+"','"+str(path)+"','"+details+"',0,'admin')")
    return '<script>alert("added
successfully");window.location="/admin_home"</script>'

@app.route("/admin_view_teachers")
def admin_view_teachers():
    db=Db()
    res=db.select("select * from teacher,login where teacher.teacher_id=login.login_id ")
    return render_template("admin/view_teachers.html", data=res)

@app.route("/block_teachers/<tid>")
def block_teachers(tid):
    db=Db()
    db.update("update login set usertype='block' where login_id='"+tid+"'")
    return redirect("/admin_view_teachers")
@app.route("/unblock_teachers/<tid>")

```

```

def unblock_teachers(tid):
    db=Db()
    db.update("update login set usertype='teacher' where login_id='"+tid+"'")
    return redirect("/admin_view_teachers")

@app.route("/admin_view_students")
def admin_view_students():
    db=Db()
    res=db.select("select * from student,login where student.student_id=login.login_id")
    return render_template("admin/view_students.html", data=res)

@app.route("/block_students/<sid>")
def block_students(sid):
    db=Db()
    db.update("update login set usertype='block' where login_id='"+sid+"'")
    return redirect("/admin_view_students")
@app.route("/unblock_students/<sid>")
def unblock_students(sid):
    db=Db()
    db.update("update login set usertype='student' where login_id='"+sid+"'")
    return redirect("/admin_view_students")

@app.route("/admin_send_notification")
def admin_send_notification():
    return render_template("admin/send_notificaion.html")
@app.route("/admin_send_notification_post",methods=['post'])
def admin_send_notification_post():
    topic=request.form['textfield']
    notification=request.form['textarea']
    db=Db()
    db.insert("insert into notification(type, date, topic, notification, sender_id)
values('admin',curdate(),'"+topic+"','"+notification+"', 0)")
    return '<script>alert("noification send");window.location="/admin_home"</script>'

@app.route("/admin_view_notification")
def admin_view_notification():
    db=Db()
    res=db.select("select * from notification")
    return render_template("admin/view_notification.html",data=res)
@app.route("/delete_notification/<dn>")
def delete_notification(dn):
    db=Db()
    db.delete("delete from notification where notification_id='"+dn+"'")
    return redirect("/admin_view_notification")

@app.route("/admin_view_reporting", methods=['get', 'post'])

```

```

def admin_view_reporting():
    if request.method=="POST":
        name=request.form['textfield']
        db = Db()
        res = db.select(
            "select report.*, student.name as sname from report, student where
report.member_id=student.student_id and report.type='student' and student.name like
'" + name + "%' union(select report.*, teacher.name as sname from report, teacher where
report.member_id=teacher.teacher_id and report.type='teacher' and teacher.name like
'" + name + "%')")
        return render_template("admin/View_reporting.html", data=res)
    else:
        db = Db()
        res = db.select("select report.*, student.name as sname from report, student where
report.member_id=student.student_id and report.type='student' union(select report.*,
teacher.name as sname from report, teacher where
report.member_id=teacher.teacher_id and report.type='teacher')")
        return render_template("admin/View_reporting.html", data=res)

@app.route("/admin_send_reply/<rid>")
def admin_send_reply(rid):
    return render_template("admin/send_reply.html", rid=rid)
@app.route("/admin_send_reply_post", methods=['post'])
def admin_send_reply_post():
    id=request.form['hid']
    reply=request.form['textarea']
    db=Db()
    db.update("update report set reply='" + reply + "', reply_date=curdate() where
report_id='" + id + "'")
    return '<script>alert("reply sented
successfully");window.location="/admin_home"</script>'

@app.route("/admin_view_feedback", methods=['get', 'post'])
def admin_view_feedback():
    if request.method == "POST":
        name = request.form['textfield']
        db = Db()
        res = db.select("select feedback.*,student.name as sname from feedback, student
where feedback.sender_id=student.student_id and feedback.type='student' and
student.name like '" + name + "%' union(select feedback.*, teacher.name as sname from
feedback, teacher where feedback.sender_id=teacher.teacher_id and
feedback.type='teacher' and teacher.name like '" + name + "%')")
        return render_template("admin/view_feedback.html", data=res)
    else:
        db = Db()
        res = db.select("select feedback.*,student.name as sname from feedback, student
where feedback.sender_id=student.student_id and feedback.type='student' union(select
feedback.*, teacher.name as sname from feedback, teacher where
feedback.sender_id=teacher.teacher_id and feedback.type='teacher')")
        return render_template("admin/view_feedback.html", data=res)

```

```
@app.route("/admin_view_feedback_post",methods=['post'])
def admin_view_feedback_post():
    sender_name=request.form['textarea']
```

```
@app.route("/admin_view_event")
def admin_view_event():
    db=Db()
    res=db.select('select * from event')
    return render_template("admin/view_event.html",data=res)
@app.route("/delete_event/<de>")
def delete_event(de):
    db=Db()
    db.delete("delete from event where event_id='"+de+"'")
    return redirect("/admin_view_event")
```

```
@app.route("/admin_update_event")
def admin_update_event():
    return render_template("admin/update_event.html")
@app.route("/admin_update_event_post",methods=['post'])
def admin_update_event_post():
    name=request.form['textfield']
    date=request.form['textfield2']
    image=request.form['fileField']
    details=request.form['textarea']
    return "ok"
```

```
@app.route("/aa")
def aa():
    return render_template("admin/admin_index1.html")
```

```
#####
@app.route("/and_Student_register", methods=['post'])
def and_Student_register():
    name=request.form['name']
    place=request.form['place']
    image=request.form['image']
    pin=request.form['pin']
    email=request.form['email']
    phone=request.form['phone']
    password = request.form['pa']
    import time
    timestr = time.strftime("%Y%m%d-%H%M%S")
```

ANDROID

```

a = base64.b64decode(image)
fh = open(static_path + "pic\\" + timestr + ".jpg", "wb")
path = "/static/pic/" + timestr + ".jpg"
fh.write(a)
fh.close()
db=Db()
lid = db.insert("insert into login(username, password, usertype) values('" + phone + "'",
"" + str(password) + "'", 'student')")
db.insert("insert into student(student_id, name, place, image, pin, email, phone)
values('"+str(lid)+"', '"+name+"', '"+place+"', '"+path+"', '"+pin+"', '"+email+"',
""+phone+"')")
return jsonify(status="ok")

```

```

@app.route("/and_Teacher_register", methods=['post'])

```

```

def and_Teacher_register():

```

```

    name=request.form['name']
    place = request.form['place']
    image = request.form['image']
    phone = request.form['phone']
    password = request.form['pa']
    import time
    timestr = time.strftime("%Y%m%d-%H%M%S")
    a = base64.b64decode(image)
    fh = open(static_path + "pic\\" + timestr + ".jpg", "wb")
    path = "/static/pic/" + timestr + ".jpg"
    fh.write(a)
    fh.close()

    db=Db()
    lid=db.insert("insert into login(username, password, usertype) values('"+phone+"',
""+str(password)+"', 'teacher')")
    db.insert("insert into teacher(teacher_id,name,place,image,phone) values
('"+str(lid)+"', '"+name+"', '"+place+"', '"+path+"', '"+phone+"')")
    return jsonify(status="ok")

```

```

@app.route("/and_login", methods=['post'])

```

```

def and_login():

```

```

    uname=request.form['uname']
    password = request.form['password']

    db = Db()
    res = db.selectOne("select * from login where username='" + uname + "' and
password='" + password + "'")
    print(res)
    if res is not None:
        if res['usertype'] == "teacher" or res['usertype'] == "student" or res['usertype'] ==
"block":
            return jsonify(status="ok", type=res['usertype'], lid=res['login_id'])
        else:
            return jsonify(status="no")

```



```

else:
    return jsonify(status="invalid")

@app.route("/teacher_view_events", methods=['post'])
def teacher_view_events():
    lid=request.form['lid']
    db=Db()
    res=db.select("select * from event where creator_id='"+lid+"' and type='teacher'")
    if len(res)>0:
        return jsonify(status="ok", data=res)
    else:
        return jsonify(status="no")

@app.route("/teacher_delete_event", methods=['post'])
def teacher_delete_event():
    eid=request.form['eid']
    db=Db()
    db.delete("delete from event where event_id='"+eid+"'")
    return jsonify(status="ok")

@app.route("/teacher_view_notification", methods=['post'])
def teacher_view_notification():
    lid=request.form['lid']
    db=Db()
    res=db.select("select * from notification where sender_id='"+lid+"' and
type='teacher'")
    if len(res)>0:
        return jsonify(status="ok", data=res)
    else:
        return jsonify(status="no")

@app.route("/teacher_delete_notification", methods=['post'])
def teacher_delete_notification():
    nid=request.form['nid']
    db=Db()
    db.delete("delete from notification where notification_id='"+nid+"'")
    return jsonify(status="ok")

@app.route("/teacher_view_groups", methods=['post'])
def teacher_view_groups():
    lid=request.form['lid']
    db=Db()
    res=db.select("select * from `group` where manager_id='"+lid+"' and
type='teacher'")
    print(res)
    if len(res)>0:
        return jsonify(status="ok", data=res)

```

```

    else:
        return jsonify(status="no")

@app.route("/teacher_delete_group", methods=['post'])
def teacher_delete_group():
    gid=request.form['gid']
    db=Db()
    db.delete("delete from `group` where group_id='"+gid+"'")
    return jsonify(status="ok")

@app.route("/teacher_add_group", methods=['post'])
def teacher_add_group():
    lid=request.form['lid']
    grpname=request.form['group']
    db=Db()
    db.insert("insert into `group` (manager_id, type, name, date) values('"+lid+"',
'teacher', '"+grpname+"', curdate())")
    return jsonify(status="ok")

@app.route("/teacher_view_student", methods=['post'])
def teacher_view_student():
    gid=request.form['gid']
    db = Db()
    res = db.select("select * from `student` where student_id not in (select stud_id from
group_member where group_id='"+gid+"')")
    print(res)
    if len(res) > 0:
        return jsonify(status="ok", data=res)
    else:
        return jsonify(status="no")

@app.route("/teacher_add_grp_member", methods=['post'])
def teacher_add_grp_member():
    gid=request.form['gid']
    sid=request.form['sid']
    db=Db()
    db.insert("insert into group_member(group_id, stud_id) values('"+gid+"',
 '"+sid+"')")
    return jsonify(status="ok")

@app.route("/teacher_add_notification", methods=['post'])
def teacher_add_notification():
    lid=request.form['lid']
    topic=request.form['topic']
    notification= request.form['notification']
    db=Db()
    db.insert("insert into `notification` (type,date,topic,notification,sender_id)
values('teacher',curdate(),'"+topic+"', '"+notification+"', '"+lid+"')")

```

```

    return jsonify(status="ok")

@app.route("/teacher_add_event", methods=['post'])
def teacher_add_event():
    lid=request.form['lid']
    name=request.form['name']
    img= request.form['img']
    details= request.form['details']
    db=Db()
    db.insert("insert into `event`(name,date,image,details,creator_id,type)
values('"+name+"',curdate(),'"+img+"','"+details+"','"+lid+"','teacher')")
    return jsonify(status="ok")

@app.route("/and_add_event", methods=['post'])
def and_add_event():
    name=request.form['name']
    date = request.form['date']
    image = request.form['image']
    details = request.form['details']
    lid = request.form['lid']

    import time
    timestr = time.strftime("%Y%m%d-%H%M%S")
    a = base64.b64decode(image)
    fh = open(static_path + "pic\\" + timestr + ".jpg", "wb")
    path = "/static/pic/" + timestr + ".jpg"
    fh.write(a)
    fh.close()

    db=Db()
    db.insert("insert into event(name,date,image,details, creator_id, type) values
('"+name+"', '"+date+"', '"+path+"', '"+details+"', '"+str(lid)+"', 'teacher')")
    return jsonify(status="ok")

@app.route("/send_feedback", methods=['post'])
def send_feedback():
    id=request.form['id']
    feedback=request.form['feedback']
    type=request.form['type']
    db=Db()
    db.insert("insert into feedback(type,sender_id,date,feedback)
values('"+type+"', '"+id+"',curdate(),'"+feedback+"')")
    return jsonify(status="ok")

@app.route("/add_report", methods=['post'])
def add_report():
    id=request.form['id']
    report=request.form['report']
    type = request.form['type']

```

```

db=Db()
db.insert("insert into report(type,member_id,date,report,reply_date,reply) values
('"+type+"','"+id+"',curdate(),'"+report+"','pending','pending')")
return jsonify(status="ok")

@app.route("/and_view_report", methods=['post'])
def and_view_report():
    lid=request.form['lid']
    db=Db()
    res=db.select("select * from report where member_id='"+lid+"'")
    return jsonify(status="ok", data=res)

@app.route("/and_delete_report", methods=['post'])
def and_delete_report():
    rid=request.form['rid']
    db=Db()
    db.delete("delete from report where report_id='"+rid+"'")
    return jsonify(status="ok")

@app.route("/teacher_share_article", methods=['post'])
def teacher_share_article():
    lid=request.form['lid']
    image = request.form['image']
    atype = request.form['atype']
    stype = request.form['stype']
    gid = request.form['gid']

    import time
    timestr = time.strftime("%Y%m%d-%H%M%S")
    a = base64.b64decode(image)
    fh = open(static_path + "pic\\" + timestr + "." + atype, "wb")
    path = "/static/pic/" + timestr + "." + atype
    fh.write(a)
    fh.close()

    db=Db()
    db.insert("insert into article(sender_id,type,article,date, group_id)
values('"+lid+"','"+stype+"','"+path+"',curdate(), '"+gid+"'")
    return jsonify(status="ok")

@app.route("/teacher_share_idea", methods=['post'])
def teacher_share_idea():
    id=request.form['id']
    ideas=request.form['ideas']
    db=Db()
    db.insert("insert into ideas(idea,teacher_id,date)
values('"+ideas+"','"+id+"',curdate())")
    return jsonify(status="ok")

```

```

@app.route("/teacher_view_profile", methods=['post'])
def teacher_view_profile():
    id=request.form['lid']
    db = Db()
    res=db.selectOne("select * from teacher where teacher_id='"+id+"'")
    return jsonify(status="ok",
name=res['name'],place=res['place'],image=res['image'],phone=res['phone'])

@app.route("/teacher_view_discussion", methods=['post'])
def teacher_view_discussion():
    db=Db()
    res=db.select("select * from discussion, student where
student.student_id=discussion.from_id order by discussion_id desc")
    return jsonify(status="ok", data=res)

@app.route("/teacher_view_approved_reply", methods=['post'])
def teacher_view_approved_reply():
    id=request.form['did']
    db=Db()
    # res=db.select("select * from student,discussion_sub where
student.student_id=discussion_sub.sender_id and discussion_sub.discussion_id='"+id+"'
and status='pending'")
    res=db.select("select * from student,discussion_sub where
student.student_id=discussion_sub.sender_id and
discussion_sub.discussion_id='"+id+"' and status='approved'")
    print(res)
    return jsonify(status="ok", data=res)

@app.route("/teacher_view_discussion_reply", methods=['post'])
def teacher_view_discussion_reply():
    id=request.form['did']
    db=Db()
    # res=db.select("select * from student,discussion_sub where
student.student_id=discussion_sub.sender_id and discussion_sub.discussion_id='"+id+"'
and status='pending'")
    res=db.select("select * from student,discussion_sub where
student.student_id=discussion_sub.sender_id and
discussion_sub.discussion_id='"+id+"' and status='pending'")
    print(res)
    return jsonify(status="ok", data=res)

@app.route("/teacher_approve_reply", methods=['post'])
def teacher_approve_reply():
    rid=request.form['rid']
    db=Db()
    db.update("update discussion_sub set status='approved' where sub_id='"+rid+"'")
    return jsonify(status="ok")

```

```

@app.route("/teacher_reject_reply", methods=['post'])
def teacher_reject_reply():
    rid=request.form['rid']
    db=Db()
    db.update("update discussion_sub set status='rejected' where sub_id='"+rid+"'")
    return jsonify(status="ok")

@app.route("/view_profile_student", methods=['post'])
def view_profile_student():
    id=request.form['lid']
    db = Db()
    res=db.selectOne("select * from student where student_id='"+id+"'")
    return jsonify(status="ok",
name=res['name'],place=res['place'],image=res['image'],phone=res['phone'],email=res['em
ail'],pin=res['pin'])

@app.route("/profile_update_student", methods=['post'])
def profile_update_student():
    name = request.form['name']
    place = request.form['place']
    phone = request.form['phone']
    email = request.form['email']
    pin = request.form['pin']
    image = request.form['attach']
    lid = request.form['lid']
    if image != '':
        import time
        timestr = time.strftime("%Y%m%d-%H%M%S")
        a = base64.b64decode(image)
        fh = open(static_path + "pic\\" + timestr + ".jpg", "wb")
        path = "/static/pic/" + timestr + ".jpg"
        fh.write(a)
        fh.close()
        db=Db()
        db.update("update student set image='"+path+"' where student_id='"+lid+"'")
        db=Db()
        db.update("update student set name='"+name+"', place='"+place+"', pin='"+pin+"',
email='"+email+"', phone='"+phone+"' where student_id='"+lid+"'")
        db.update("update login set username='"+email+"' where login_id='"+lid+"'")
        return jsonify(status="ok")

@app.route("/view_notification", methods=['post'])
def view_notification():
    db=Db()
    res=db.select("select * from notification order by notification_id desc")
    if len(res)>0:
        return jsonify(status="ok", data=res)
    else:

```

```

    return jsonify(status="no")

@app.route("/view_events", methods=['post'])
def view_events():
    db=Db()
    res=db.select("select * from event order by event_id desc")
    if len(res) > 0:
        return jsonify(status="ok", data=res)
    else:
        return jsonify(status="no")

@app.route("/student_view_groups", methods=['post'])
def student_view_groups():
    db=Db()
    lid=request.form['lid']
    res=db.select("select * from `group`,group_member where
group.group_id=group_member.group_id and stud_id='"+lid+"'")
    print(res)
    if len(res) > 0:
        return jsonify(status="ok", data=res)
    else:
        return jsonify(status="no")

@app.route("/teacher_delete_grp_member", methods=['post'])
def teacher_delete_grp_member():
    mid=request.form['mid']
    db=Db()
    db.delete("delete from group_member where group_member_id='"+mid+"'")
    return jsonify(status="ok")

@app.route("/teacher_view_group_members", methods=['post'])
def teacher_view_group_members():
    gid=request.form['gid']
    db=Db()
    res=db.select("select student.*, group_member.group_member_id from student,
group_member where group_member.stud_id=student.student_id and
group_member.group_id='"+gid+"'")
    return jsonify(status="ok", data=res)

@app.route("/stud_view_article", methods=['post'])
def stud_view_article():
    lid=request.form['lid']
    db=Db()
    res=db.select("select article.*, student.name as name from article,student where
article.sender_id=student.student_id and article.sender_id!='"+lid+"' union (select
article.*, teacher.name as name from article,teacher where
article.sender_id=teacher.teacher_id and article.sender_id!='"+lid+"')")
    return jsonify(status="ok", data=res)

```

```

@app.route("/student_delete_article", methods=['post'])
def student_delete_article():
    aid=request.form['aid']
    db=Db()
    db.delete("delete from article where article_id='"+aid+"'")
    return jsonify(status="ok")

```

```

@app.route("/stud_view_article_own", methods=['post'])
def stud_view_article_own():
    # lid=request.form['lid']
    gid=request.form['gid']
    db=Db()
    res=db.select("select * from article where group_id='"+gid+"'")
    return jsonify(status="ok", data=res)

```

```

@app.route("/insert_discussion", methods=['post'])
def insert_discussion():
    lid=request.form['lid']
    topic=request.form['topic']
    db=Db()
    db.insert("insert into discussion(topic, date, from_id) values('"+topic+"', curdate(), '"+lid+"')")
    return jsonify(status="ok")

```

```

@app.route("/student_view_discussion", methods=['post'])
def student_view_discussion():
    db=Db()
    res=db.select("select * from discussion, student where student.student_id=discussion.from_id")
    return jsonify(status="ok", data=res)

```

```

@app.route("/stud_view_ideas", methods=['post'])
def stud_view_ideas():
    db=Db()
    res=db.select("select * from ideas, teacher where teacher.teacher_id=ideas.teacher_id order by ideas_id desc")
    return jsonify(status="ok", data=res)

```

```

@app.route("/and_like_idea", methods=['post'])
def and_like_idea():
    idea_id=request.form['id']
    lid=request.form['lid']
    db=Db()
    res=db.selectOne("select * from `like` where idea_id='"+idea_id+"' and person_id='"+lid+"'")
    if res is not None:

```



```

        return jsonify(status="no")
    else:
        db=Db()
        db.insert("insert into `like` values(null, '"+idea_id+"', '"+lid+"', curdate())")
        return jsonify(status="ok")

@app.route("/student_view_discussion_reply", methods=['post'])
def student_view_discussion_reply():
    id=request.form['did']
    db=Db()
    res=db.select("select * from student_discussion_sub where
student.student_id=discussion_sub.sender_id and
discussion_sub.discussion_id='"+id+"' and status='approved'")
    return jsonify(status="ok", data=res)
@app.route("/insert_answer", methods=['post'])
def insert_answer():
    lid=request.form['lid']
    did=request.form['did']
    ans=request.form['ans']
    db=Db()
    db.insert("insert into discussion_sub(discussion_id, sender_id, content, date, status)
values('"+did+"', '"+lid+"', '"+ans+"', curdate(), 'pending')")
    return jsonify(status="ok")

@app.route("/in_message2", methods=['post'])
def in_message2():
    lid = request.form['fid']
    toid = request.form['toid']
    msg = request.form['msg']
    db=Db()
    db.insert("insert into chat(date,from_id,to_id,message)
values(curdate(),'"+lid+"', '"+toid+"', '"+msg+"')")
    res=db.select("select * from question")
    qns=[]
    answers=[]
    scores=[]
    from check import chk
    obj=chk()
    for i in res:
        qn=i['question']
        qns.append(i['question'])
        answers.append(i['answer'])
        vec1=obj.text_to_vector(msg)
        vec2=obj.text_to_vector(qn)
        cosine = obj.get_cosine(vec1, vec2)
        scores.append(cosine)
    print("Score : ", scores)
    max_score=max(scores)
    print("Max : ", max_score)

```

```

idx=scores.index(max_score)
print("Index :", idx)
print("Question : ", qns[idx])
print("Answer : ", answers[idx])
answer = answers[idx]
if max_score == 0.0:
    answer = "Sorry. I didnt understand your question."

db = Db()
db.insert(
    "insert into chat(date,from_id,to_id,message) values(curdate(),'' + toid + ',' + lid
+ ',' + answer + ')")
    return jsonify(status="ok")

@app.route("/view_message2", methods=['post'])
def view_message2():
    fid=request.form['fid']
    toid=request.form['toid']
    lastmsgid=request.form['lastmsgid']
    print(fid, toid, lastmsgid)
    db=Db()
    res=db.select("select * from chat where ((from_id='"+fid+"' and to_id='"+toid+"' ) or
(from_id='"+toid+"' and to_id='"+fid+"' ) and chat_id>"+lastmsgid+"")
    print(res)
    return jsonify(status="ok", data=res)

if __name__ == '__main__':
    app.run(port=4000, host="0.0.0.0")

```

## **6. SYSTEM TESTING**

## 6.1 TESTING AND EVALUATION

Testing is a process of executing a program with the intent of finding an error. Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design and coding. Testing includes verifications of the basic logic of each program and verification that the entire system works properly. Testing demonstrates that software functions appear to be working according to specification. In addition, data collected as testing is conducted provides a good indication of software quality as a whole. The debugging process is the most unpredictable part of testing process. Testing begins at the module level and works towards the integration of the entire computer-based system. Testing and debugging are different activities, but any testing includes debugging strategy for software testing must accommodate low level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system function, against customer requirements. No testing is complete without verification and validation part. The goals of verification and validation activities are to assess and improve the quality of work products generated during the development and modification of the software. There are two types of verification: life cycle verification and formal verification. Life cycle verification is the process of determining the degree to which the products of the given phase of the development cycle fulfil the specification established during the prior process. Formal verification is the rigorous mathematical demonstration that source code conforms to its specifications. Validation is a process of evaluating software at the end of the software development process to determine conformance with the requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The primary objectives, when we test software are the following:

- Testing is a process of executing with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.
- A successful test is one uncovers undiscovered errors.

Thus, testing plays a very critical role in determining the reliability and efficiency of the software and hence is very important stage in software development. Tests are to be conducted on the software to evaluate its performance under a number of conditions. Ideally, it should do so at the level of each module and also when all of them are integrated to form the completed system. Software testing is done at different levels.

## 6.2 TESTING STRATEGIES

A strategy for software testing integrates software test case design method in to a well-planned series of steps that result in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote a customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project progresses.

**The general characteristics of software testing strategies are:**

- Testing begins at the component level and works “outward” toward the integration of the entire computer system.
- Different testing techniques are appropriate at different points in time.

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

A strategy must provide guidance for the practitioner and set of mild stones for the manager. Because the step on the test strategy occurs at a time when deadline pressure begins to rise, progress must be measurable and problem must surface as early as possible.

The software team’s approach to testing is defining a plan that describes an overall strategy and a procedure that defines specific testing steps and tests that will be conducted. In the proposed system, if the administrator makes any attempt to login to the application without entering his password, then the system will not allow the user to login to the application.

## 6.3 TESTING TECHNIQUES

The various testing techniques are given below:

### 6.3.1 WHITE BOX TESTING

White-box testing is also called as glass-box testing, is a test case design method that goes to the control structure of the procedural design to derive

test cases. Using white box testing methods, the software engineer can derive test cases that,

- ✓ Guarantee that all independent paths within a module have been exercised at Least once.
- ✓ Exercise all logical decision on their true and false sides.
- ✓ Execute all loops at their boundaries and within their operational sides.
- ✓ Exercise internal data structure to ensure their validity.

White box testing was successfully conducted on our system. All independent paths within a module have been executed at least once and all logical decisions have been exercised on their true and false sides.

### 6.3.2 BLACK BOX TESTING

Black-box testing is also called as behavioural testing, focuses on the functional requirement of the software. It is a complimentary approach that is likely uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories.

- ✓ Incorrect or missing functions.
- ✓ Interface errors.
- ✓ Error on data structures or external database access.
- ✓ Behaviour or performance errors.
- ✓ Initialization and termination errors.

Black box testing was successfully conducted on your system. The system was divided into a number of modules and testing was conducted on each module. We have tested the system for incorrect or missing functions, interface and performance errors.

### **6.3.3 UNIT TESTING**

Unit testing comprises the set of tests performed by an individual programmer prior to the integration of the system. Testing removes residual bugs and improves the reliability of the system.

Testing allows the developer to find out the design faults if any, and enable correction if needed. Exhaustive unit testing has to be carried out to ensure the validity of the data. In order to successfully test the entire package, unit testing is carried out. Each module was tested as when it was developed. Thus, it proved easier to conduct minute testing operation and correct them then and there.

### **6.3.4 INTEGRATION TESTING**

Bottom-up integration is the traditional strategy used to integrate the component of a software system into a functional whole. Bottom-up integration consists of unit testing, followed by subsystem testing and followed by testing of the entire system. Unit testing has the goal of discovering errors in the individual parts of the system.

Parts are tested in isolation from one another in an artificial environment known as “Test Harness”, which consists of driver programs and data necessary to exercise the modules. Unit testing should be as exhaustive as possible to ensure that each representative case handled by each module has been tested. Unit testing is eased by a system structure that is composed of small loosely coupled modules.

Both control and data interfaces must be tested. Large software system may require several levels of subsystem testing. Lower level subsystems are successively combined to form higher level subsystems. In most software systems, exhaustive testing of subsystem capabilities is not feasible due to the combination complexity of the module interfaces. Therefore, test cases must be carefully chosen to exercise the interfaces in the desired manner.

### **6.3.5 ACCEPTANCE TESTING**

Acceptance testing involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements. It is not unusual for two sets of acceptance test to be run, those developed by the quality group and those developed by the customer.

In addition to the functional and performance tests, stress tests are performed to determine the limitation of the system. For example, a compiler might be tested to determine the effect of the symbol table overflow, or real-time system might be tested to determine the effect of simultaneous arrival of numerous high priority interrupts.

### **6.3.6 OUTPUT TESTING**

Output testing of the proposed system is important since no system could be useful if it does not produce the required output.

The output format on the screen is found to be correct, as the format was designed in the system design phase according to the user needs. For the hard copy also the output comes out as the specified requirements by the user. Hence output testing doesn't result in any correction on the system.



## **7.SYSTEM IMPLEMENTATION AND DEPLOYMENT**

Implementation is the process of deploying the new system in the operational environment. Proper implementation is essential to provide a reliable system to meet the organizational requirement. There are four types of implementation methods. They are Direct Changeover, Phased Implementation, Parallel Run and Pilot Approach. The most commonly used implementation methods are Pilot Approach and Parallel Run.

The system which is developed as a web application hence the other functions as normal application, as usual some web development technologies are used in the implementation of the project. The language I selected to program this software is PHP. The reason I selected PHP is that is a simple and powerful language that especially developed to create web application.

Technologies used in the development of the software are:

- ✓ Programming language – Python
- ✓ DBMS – MySQL server
- ✓ Development tool – PyCharm
- ✓ Development platform – Windows 10

The front end is HTML, and CSS and back end is MySQL Server and Python. The system developed on PyCharm in Windows 10 operating system.

## **8. CONCLUSION**

The ‘CampusTalk: your personal college chatbot’ has been developed for all given conditions and it is found working effectively under the all the circumstances that may arise in the real environment. It is helpful in guiding students with correct and most up to date sources of information. It is advantageous for international applicants for queries such as fee payment and academic matters. Students can get the information at their fingertips rather than visiting college office. It improves efficiency by taking over tasks for which humans are not essential. Nevertheless, active learning helps to improve the bot performance for handling off-script queries. The system is done with an insight into the necessary modifications that may be required in the future. Hence the system can be maintained successfully without much work.

## **8.1 FUTURE ENHANCEMENT**

To improve the current functionalities of College Chatbot, in the future, the scope of the chatbot can be increased by inserting data for all the departments, training the bot with varied data, testing it on live website, and based on that feedback inserting more training data to the bot. Some of the new features which can be added to the bot are

- 1) speech recognition feature through which students can ask their queries verbally and get the answers from the bot.
- 2) integration with multiple channels such as phone call, SMS, and various social media platforms like Skype, Facebook and Twitter.
- 3) handling context aware and interactive queries in which bot will be aware of the context of an ongoing conversation with a student.
- 4) integration with services such as password reset and course enrolment.

## **9.REFERENCE**

### **BOOKS:**

Database System Concept: Marvin .F. Korth

A Byte of Python : Swaroop C H

Apress Software Engineering : Rajib Mall

### **WEBSITES:**

[www.wikipedia.com](http://www.wikipedia.com)

[www.tutorialspoint.com](http://www.tutorialspoint.com)

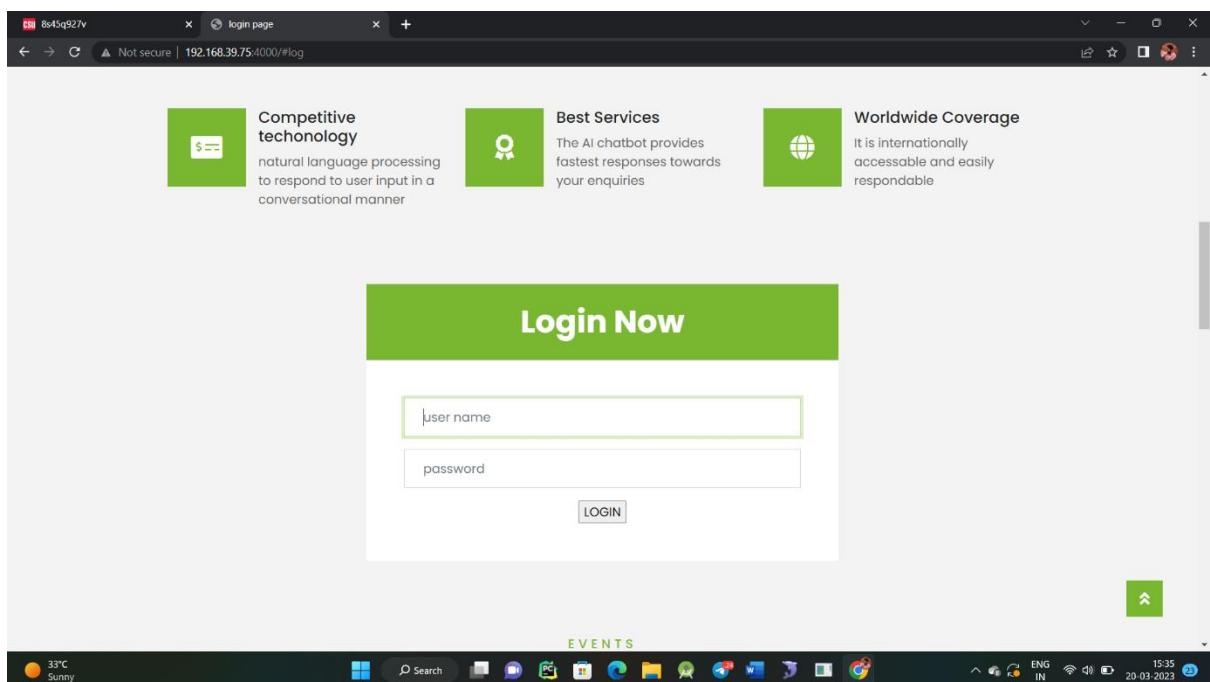
[www.youtube.com](http://www.youtube.com)

[www.codementor.io](http://www.codementor.io)

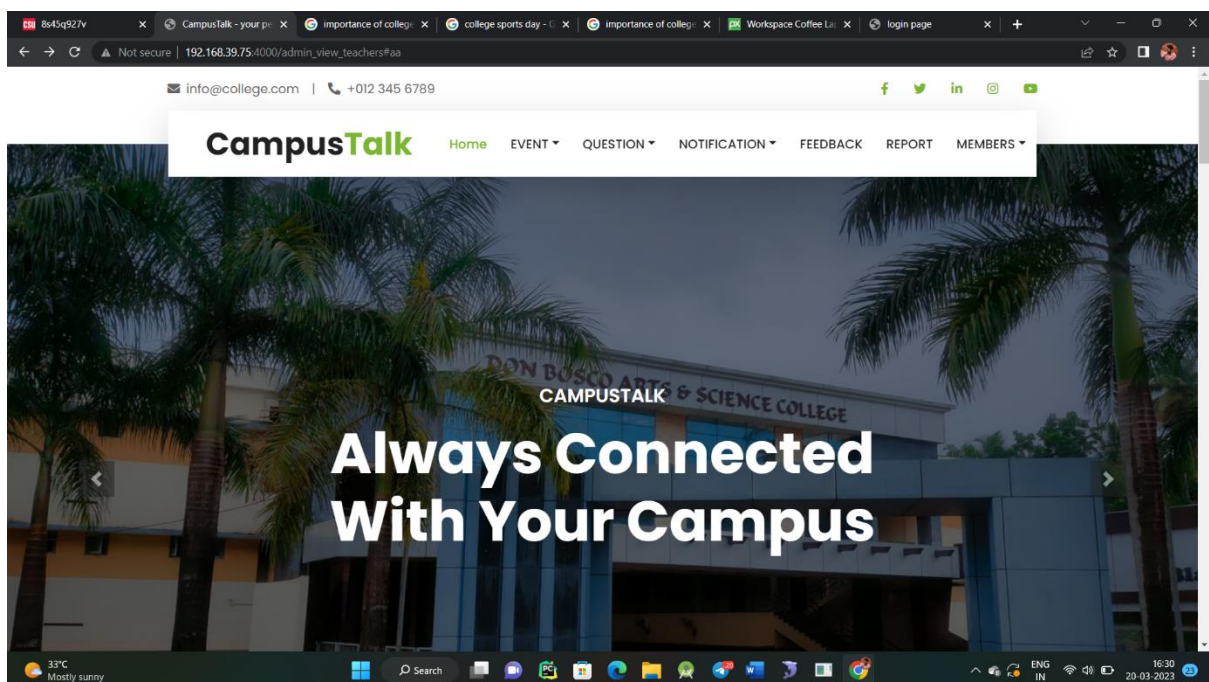
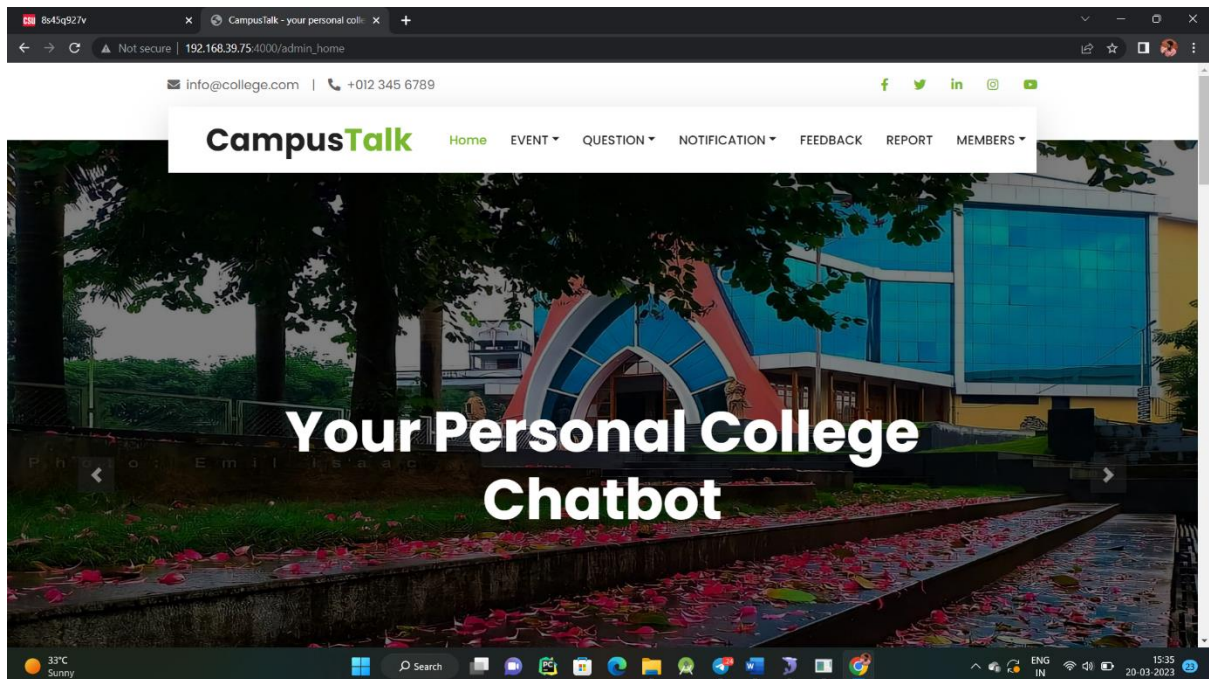
## **10. APPENDIX**

## SCREENSHOTS

Admin-login page:



## Admin- home page



8s45q92/v CampusTalk - your personal coll... 192.168.39.75:4000/admin\_home

## EVENTS

### Our Latest Events

**COMET 6**  
NATIONAL COMMERCE FEST  
JANUARY 13, 14 2023  
POWERED BY Anix

COMET | NATIONAL COMMERCE FEST

**EQUINOX**  
DARE TO BE JUDGED  
2020  
SOUTH INDIAN MANAGEMENT FEST  
08<sup>TH</sup> FEBRUARY 2020

EQUINOX | SOUTH INDIAN MANAGEMENT FEST

**RESSAISIR**  
National Conference on  
"Emerging Health Issues and Social Work Profession"  
March 10, 11

RESSAISIR | SOCIAL WORKERS NATIONAL CONFERENCE

33°C Sunny 20-03-2023 15:35

8s45q92/v CampusTalk - your personal coll... 192.168.39.75:4000/admin\_home

## TESTIMONIAL

### What Say Our Users

The college chatbot was a lifesaver for me! Whenever I had a question about a deadline or needed help finding a resource, the chatbot was always available to provide quick and accurate information.

**Adarsh Joe**  
student

As a faculty member, I found the college chatbot to be a great way to help students stay on track with their coursework. I could program the chatbot to send reminders about upcoming assignments or direct students to additional resources, which was incredibly helpful for both me and my students.

**Hebin Layola**  
Teacher

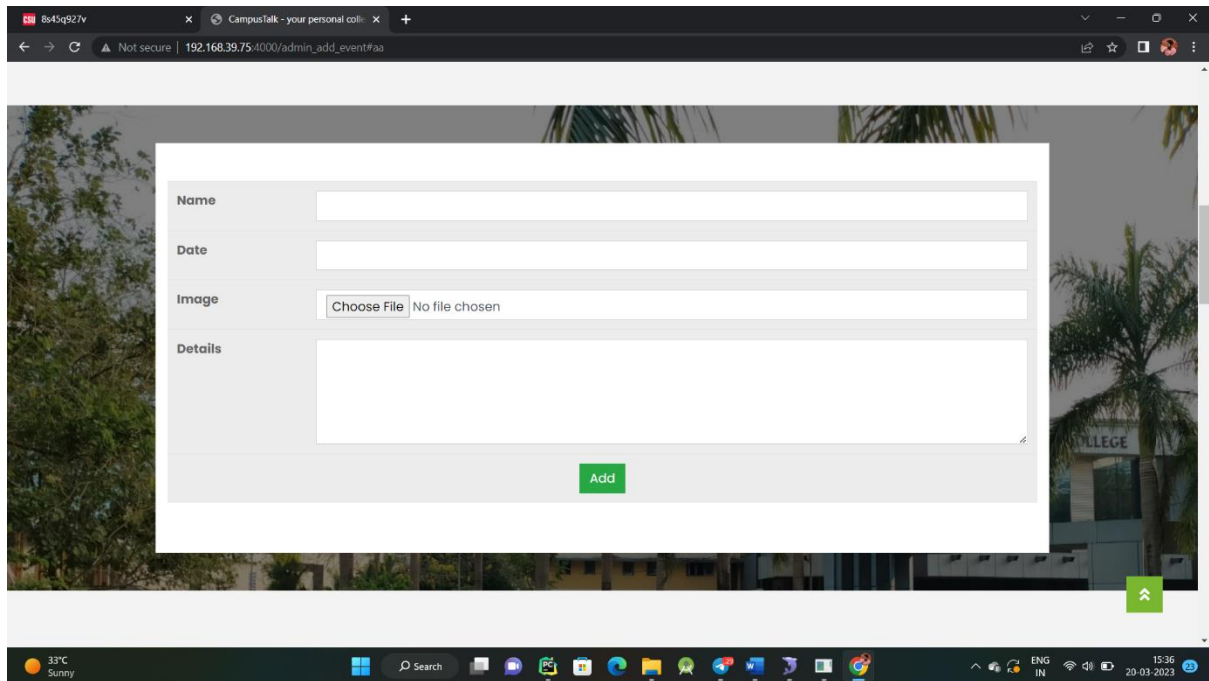
I was skeptical about using a chatbot at first, but I was pleasantly surprised by how helpful and efficient it was. I didn't have to wait on hold or navigate a confusing website - the chatbot had all the information I needed right at my fingertips.

**Sony Mariya**  
student

33°C Sunny 20-03-2023 15:35



## Add event



8s4bq927v CampusTalk - your personal colli- x +

Not secure | 192.168.39.75:4000/admin\_add\_event#aa

Name

Date

Image  No file chosen

Details

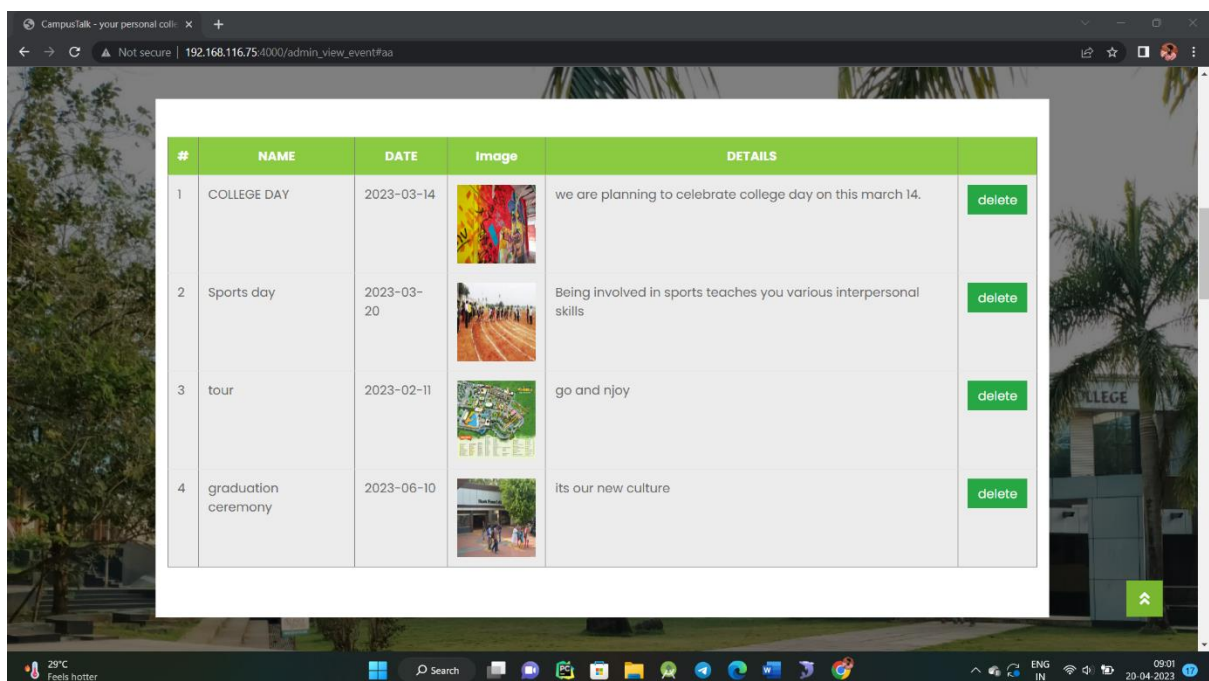
Add

33°C Sunny

Search





ENG IN 15:36 20-03-2023

## View event



CampusTalk - your personal colli- x +

Not secure | 192.168.116.75:4000/admin\_view\_event#aa

#	NAME	DATE	Image	DETAILS	
1	COLLEGE DAY	2023-03-14		we are planning to celebrate college day on this march 14.	<input type="button" value="delete"/>
2	Sports day	2023-03-20		Being involved in sports teaches you various interpersonal skills	<input type="button" value="delete"/>
3	tour	2023-02-11		go and njoy	<input type="button" value="delete"/>
4	graduation ceremony	2023-06-10		its our new culture	<input type="button" value="delete"/>

29°C Feels hotter

Search

ENG IN 09:01 20-04-2023

## Add questions

Question

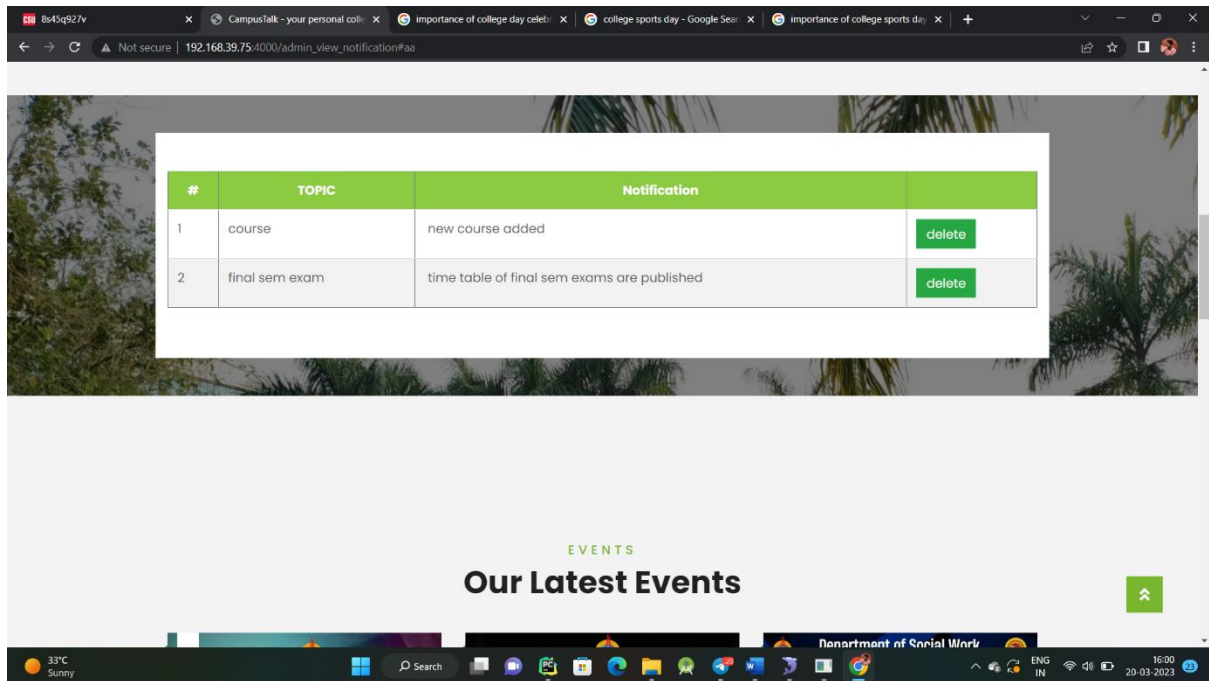
Answer

Add

## View questions

#	QUESTION	ANSWER	
1	what is the name of college?	Don Bosco Arts & Science College,Angadikadavu	Delete
2	who is the principal?	Dr. Francis Karackat SDB	Delete
3	total number of courses	16	Delete
4	why do i chose this college	donBosco college is declared as a model college by the Kannur University Syndicate.Recipient of University awards for the best NSS Unit, best NSS Programme Officer and best NSS Volunteer in 2017-2018 and 2019-2020 academic years.	Delete
5	which are the commerce based courses available?	UG couoses -bcom finance ,bcom computer ,bcom coorporation,PG courses-Mcom finance	Delete
6	which are the available digree programs?	bcom finance ,bcom computer ,bcom coorporation,bsw,bba,bca,bsc mathematics,bsc psychology,ba english	Delete
7	which are the	mcom finance,msw,msc psychology,msc statistics,mca,mcj	Delete

## Add notification



#	TOPIC	Notification	
1	course	new course added	<button>delete</button>
2	final sem exam	time table of final sem exams are published	<button>delete</button>

EVENTS

### Our Latest Events

Department of Social Work

33°C Sunny

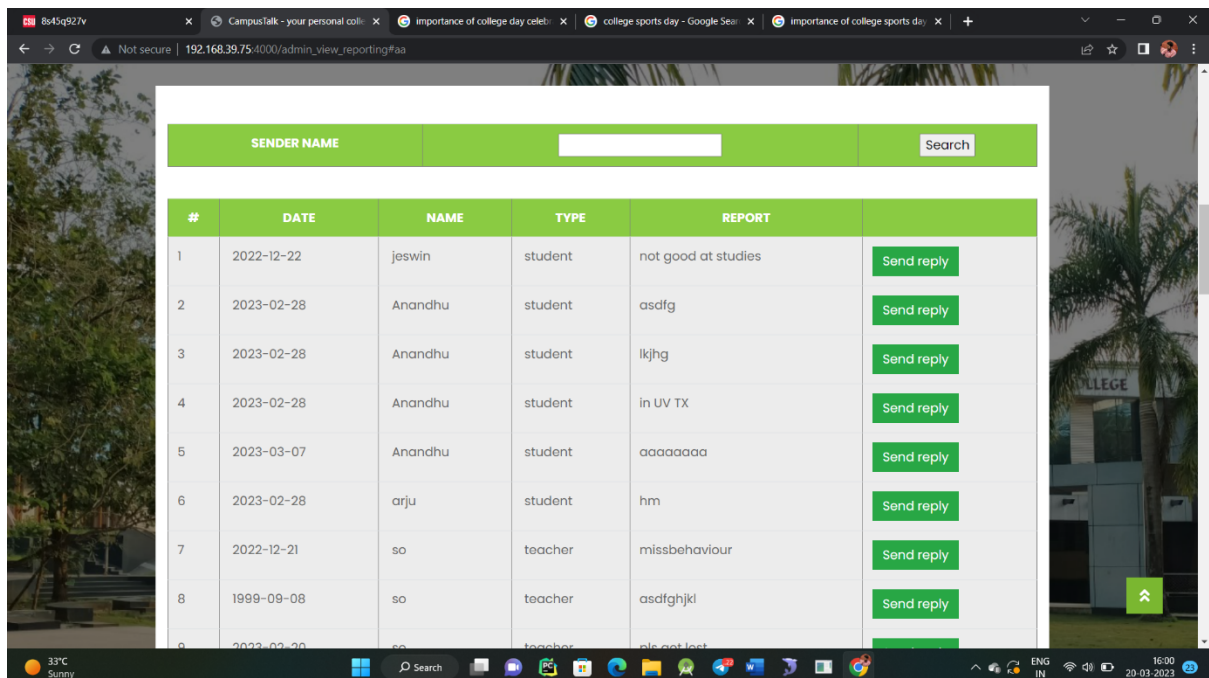
Search

ENG IN

16:00

20-03-2023

## View report & send reply



SENDER NAME

Search

#	DATE	NAME	TYPE	REPORT	
1	2022-12-22	jeswin	student	not good at studies	<button>Send reply</button>
2	2023-02-28	Anandhu	student	asdfg	<button>Send reply</button>
3	2023-02-28	Anandhu	student	lkjhg	<button>Send reply</button>
4	2023-02-28	Anandhu	student	in UV TX	<button>Send reply</button>
5	2023-03-07	Anandhu	student	aaaaaaa	<button>Send reply</button>
6	2023-02-28	arju	student	hm	<button>Send reply</button>
7	2022-12-21	so	teacher	missbehaviour	<button>Send reply</button>
8	1999-09-08	so	teacher	asdfghjkl	<button>Send reply</button>
9	2023-02-20	so	teacher	ple get lost	<button>Send reply</button>

33°C Sunny

Search

ENG IN




16:00

20-03-2023

## View Teachers

CampusTalk - your personal colli x +

Not secure | 192.168.116.75:4000/admin\_view\_teachers#aa

#	NAME	PLACE	IMAGE	PHONE	
1	Naveen	Charal		1234567890	block
2	Bixon	iritty		1212121212	block
3	Sony	kottiyoor		8590668032	block

29°C Partly sunny

Search




ENG IN

09:31 20-04-2023

## View Students

CampusTalk - your personal colli x +

Not secure | 192.168.116.75:4000/admin\_view\_students#aa

#	NAME	PLACE	IMAGE	PIN	EMAIL	PHONE	
1	Jeswin	payyavoor		670633	jeswin312003@gmail.com	8086064661	unblock
2	Alb	kolihattu		666777	alb@gmail.com	1010101010	block
3	Max	USA		606060	madmax@gmail.com	8590606269	block

29°C Partly sunny

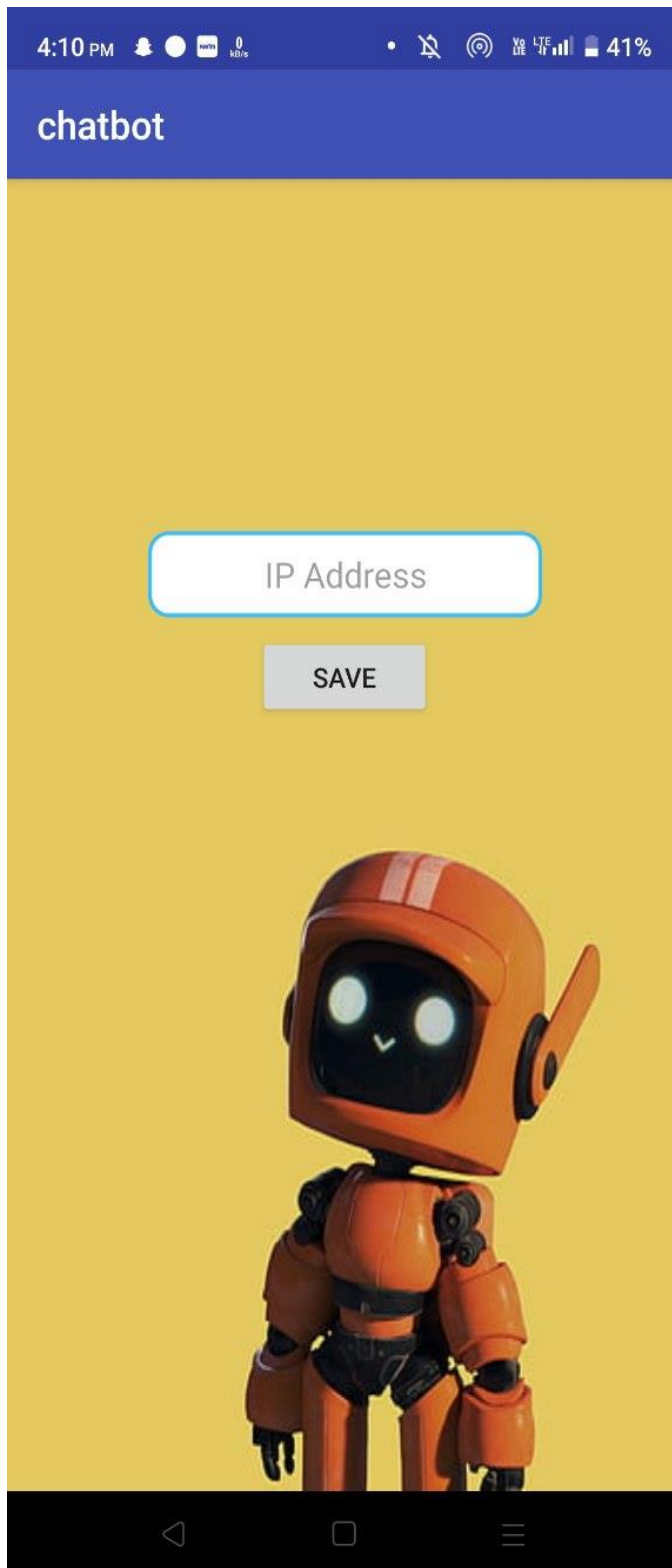
Search

ENG IN

09:31 20-04-2023

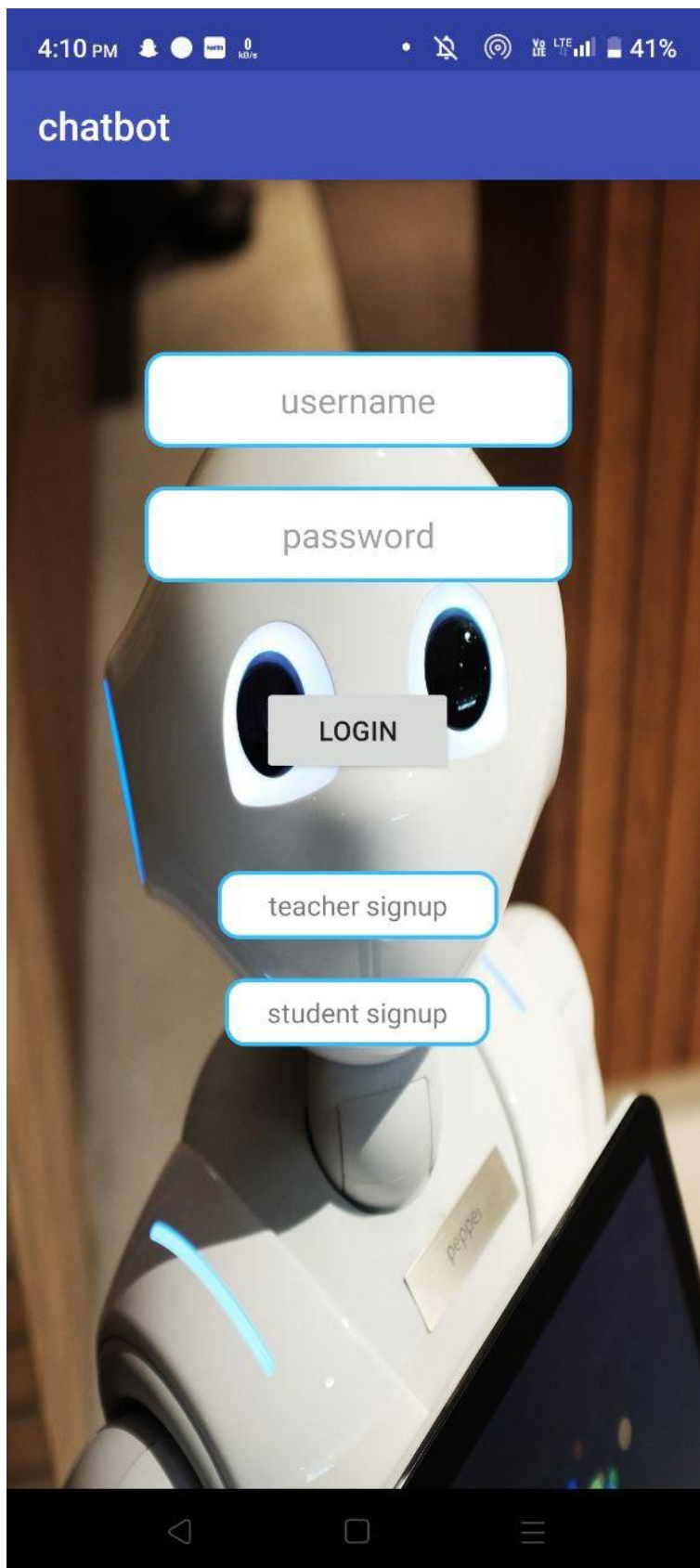
## ANDROID

User IP connection














## Login page



## Teacher signup

10:54 AM     •   Vo LTE   64%

chatbot



name

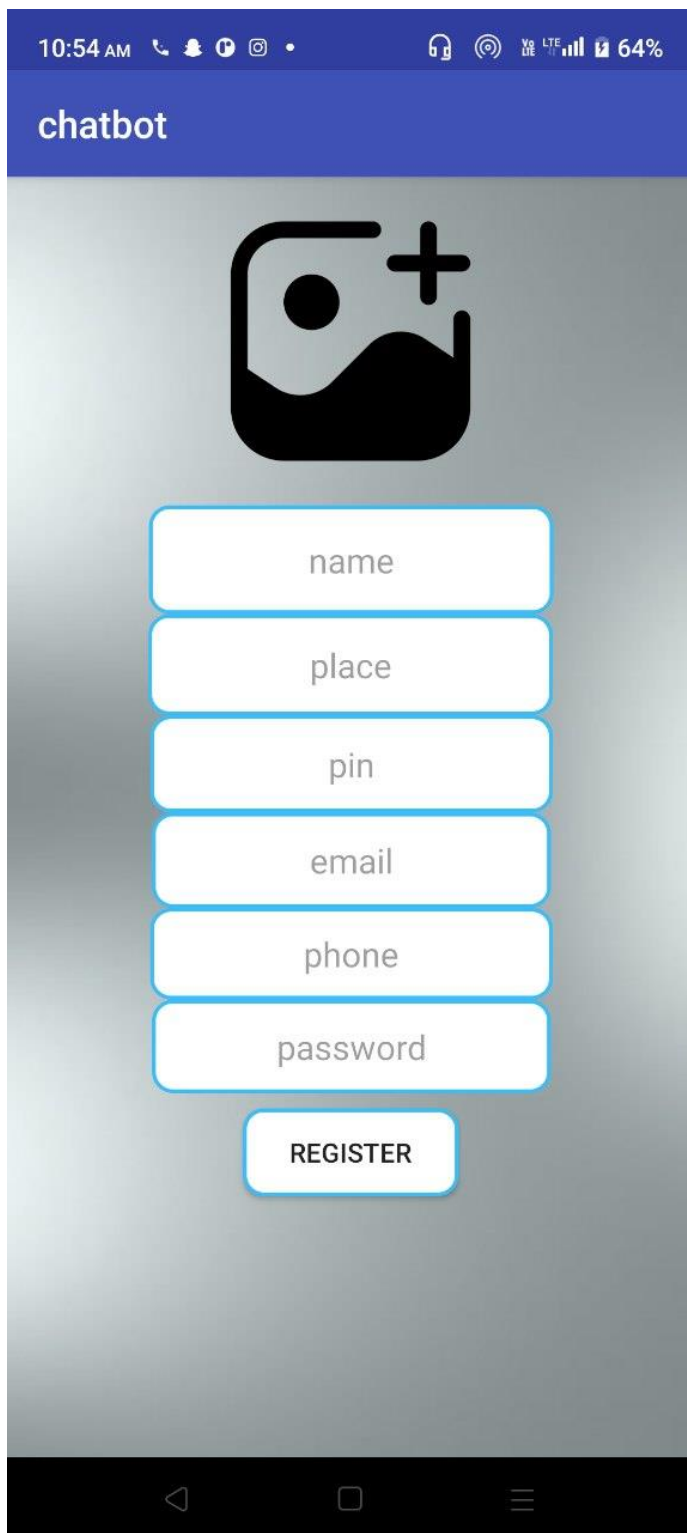
place

phone

password

REGISTER


## Student signup



The image shows a mobile application interface for student signup. At the top, a blue header bar contains the text "chatbot". Below the header, there is a large black icon of a camera with a plus sign, indicating a photo upload feature. Underneath the icon, there are six white input fields with blue borders, each containing a placeholder text: "name", "place", "pin", "email", "phone", and "password". Below these fields is a white button with a blue border and the text "REGISTER". The background of the app is a light gray gradient. The top status bar shows the time as 10:54 AM, along with various icons for notifications, social media, and battery level (64%). The bottom navigation bar is black with three white icons: a back arrow, a home square, and a menu hamburger icon.

10:54 AM

chatbot



name

place

pin

email

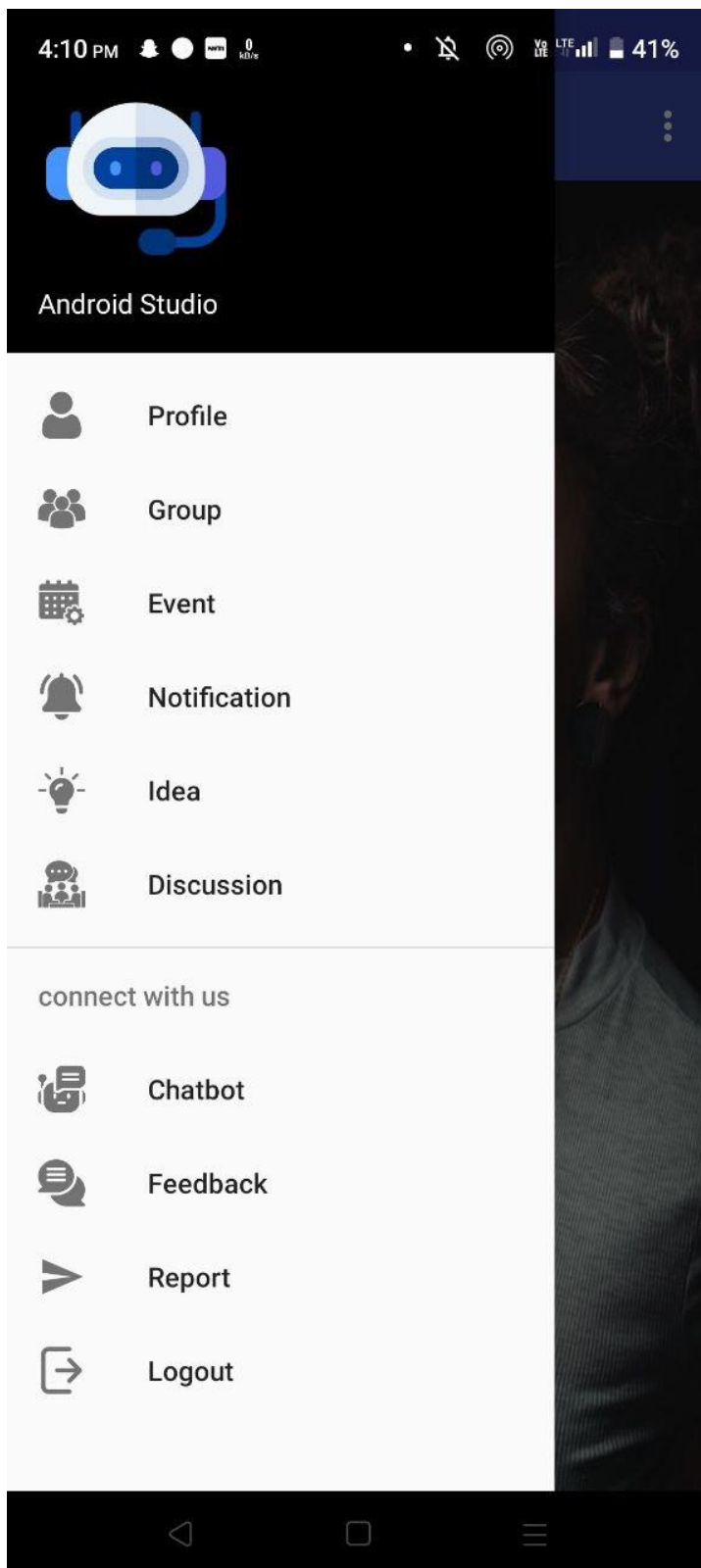
phone

password

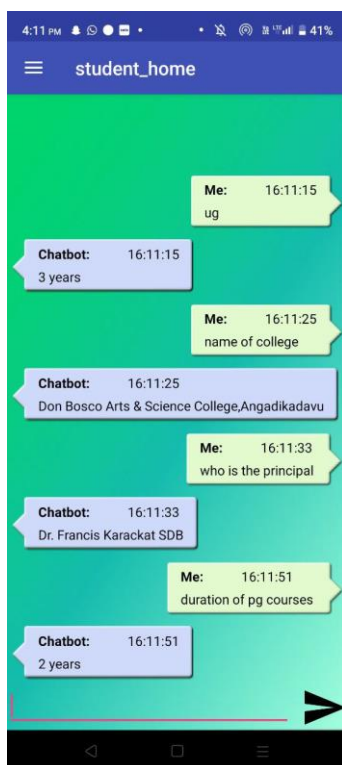
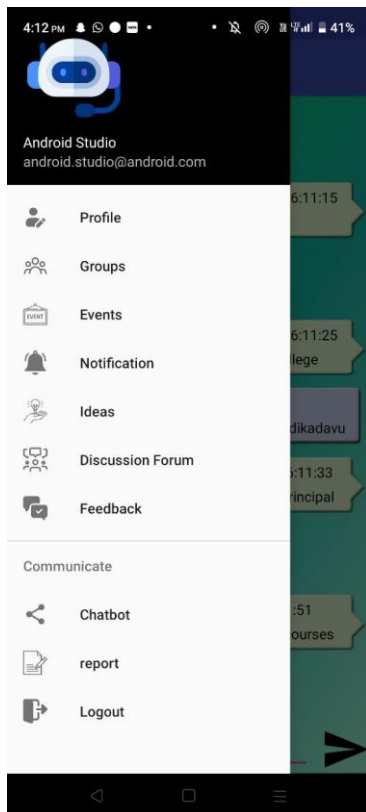
REGISTER



## Teacher home



## Student home





**A PROJECT REPORT ON**  
**CYBERRELY**  
**PHISHING WEBSITE DETECTION WITH**  
**GRAPHICAL PASSWORD**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**VISMAYA SANIL**

**REG.NO: DB20BCAR16**

**JESWIN JAMES**

**REG.NO: DB20BCAR09**

**EBY MATHEW**

**REG.NO: DB20BCAR29**

**UNDER THE SUPERVISION AND GUIDANCE OF**  
**Mr. HEBIN LAYOLA**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**

**A PROJECT REPORT ON**  
**CYBERRELY**  
**PHISHING WEBSITE DETECTION WITH**  
**GRAPHICAL PASSWORD**

Submitted in partial fulfilment of the requirement for award of the degree

Of

**Bachelor of Computer Application**

Of

**KANNUR UNIVERSITY**

By

**VISMAYA SANIL**

**REG.NO: DB20BCAR16**

**JESWIN JAMES**

**REG.NO: DB20BCAR09**

**EBY MATHEW**

**REG.NO: DB20BCAR29**

**UNDER THE SUPERVISION AND GUIDANCE OF**

**Mr. HEBIN LAYOLA**



**DON BOSCO ARTS & SCIENCE COLLEGE**

**ANGADIKADAVU, KANNUR, 670706**

**2023**

**DON BOSCO ARTS & SCIENCE COLLEGE**  
**ANGADIKADAVU**  
**IRITTY, KANNUR**



**CERTIFICATE**

Certified that this report titled **CYBERRELY-PHISHING WEBSITE DETECTION WITH GRAPHICAL PASSWORD** is a bonafide record of the project work done by **VISMAYASANIL (Reg.No:DB20BCAR16)**, **JESWIN JAMES (Reg.No:DB20BCAR09)**, **EBY MATHEW (Reg.No:DB20BCAR29)** under the supervision and guidance, towards partial fulfilment of the requirement for award of the Degree of Bachelor of Computer Application (BCA) of the Kannur University.

Project Guide

**Mr.HEBIN LAYOLA**

Head of the Department

**Mrs. SINDHU P.M**

Angadikadavu

External Examiner

Date:

1.

2.

## **Declaration**

We **VISMAYA SANIL, JESWIN JAMES and EBY MATHEW** sixth semester BCA students of Don Bosco Arts & Science College, Angadikadavu, under Kannur University do hereby declare that the project entitled **“CYBERRELY-PHISHING WEBSITE DETECTION WITH GRAPHICAL PASSWORD”** is the original work carried out by us in the sixth semester under the supervision of **Mr. HEBIN LAYOLA**, Lecture of the Department of BCA, Don Bosco Arts & Science College, Angadikadavu, in partial fulfilment of the requirement for the award of the Degree Bachelor of Computer Application, Kannur University.

ANGADIKADAVU

DATE:

VISMAYA SANIL

JESWIN JAMES

EBY MATHEW

## ACKNOWLEDGEMENT

First of all we thank the lord almighty for his immense grace and blessings showered on us at every stages of this work. We are greatly indebted to our Principal **Fr. Dr. Francis Karackat SDB**, Don Bosco Arts & Science College, Angadikadavu for providing the opportunity to take up this project as part of our curriculum.

We deeply indebted to our project guide **Mr. Hebin Layola**, lecture of department of BCA, for his assistance and valuable suggestions as guide. He made this project a reality.

We express our sincere thanks to Mrs. Sindu PM, Mr. Hebin Layola, Mrs. Fincy Cyriac, Mrs. Sruthi N and Mrs. Vineetha Mathew, lecturers of department of BCA, for their valuable suggestions during the course of this project. Their critical suggestions helped us to improve the project work.

Acknowledging the efforts of everyone, their chivalrous help in the course of the project preparation and their willingness to collaborate with the work, their magnanimity through lucid technical details lead to the successful completion of our project.

We would like to express our sincere thanks to all our friends, colleagues, parents and all those who have directly or indirectly assisted during this work.



# CONTENTS

<b>Chapters</b>	<b>Contents</b>	<b>Page No</b>
1	Introduction	1
1.1	Waterfall model	3
2	System Analysis	7
2.1	Existing System	8
2.1.1	Disadvantages of Existing System	8
2.2	Proposed System	8
2.3	Phishing Website Features	9
2.4	Random Forest Algorithm	16
2.5	Feasibility Study	19
2.5.1	Economic Feasibility	20
2.5.2	Technical Feasibility	20
2.5.3	Behavioural Feasibility	21
2.6	System Specification	21
2.6.1	Software Specification	22
2.6.2	Hardware Specification	22
2.7	Use Case Diagram	23
2.7.1	Identification of Actors	23
2.7.2	Identification Of Use Cases	23
2.7.3	Use Cases for The Actor ADMIN	24
2.7.4	Use Cases for The Actor USER	24
2.7.5	Use Case Diagram	26
3	System Design	29
3.1	Introduction	30

3.2	Database Design	30
3.3	Table Design	30
3.4	Phishing Website Detection With Graphical Password	32
3.5	Data Flow Diagram	34
3.6	ER Diagram	42
4	Coding	44
4.1	Input Interface	45
4.2	Output Interface	45
4.3	Software Description	45
4.3.1	HTML	46
4.3.2	CSS	48
4.3.3	JavaScript	51
4.3.4	Android Studio	56
4.3.5	Java	56
4.3.6	Python	59
4.3.7	MySQL	59
4.3.8	PyCharm	60
4.3.9	Flask	61
5	Coding Pages	62
5.1	Admin Page	63
5.2	User Page	68
6	System Testing	69
6.1	Testing And Evaluation	70
6.2	Testing Strategies	71

6.3	Testing Techniques	72
6.3.1	White Box Testing	72
6.3.2	Black Box Testing	72
6.3.3	Unit Testing	73
6.3.4	Integration Testing	73
6.3.5	Acceptance Testing	73
6.3.6	Output Testing	74
7	System Implementation And Deployment	75
8	Conclusion	76
8.1	Future Enhancement	76
9	Reference	77
10	Appendix	78
10.1	Screenshots	79

# **CHAPTER I**

## **INTRODUCTION**

## INTRODUCTION

The project allows user to input an image as its password and only user knows what the image looks like as a whole. On receiving the image, the system segments the image into an array of images and stores them accordingly. The next time user logs on to the system the segmented image is received by the system in a jumbled order. Now if user chooses the parts of image in an order so as to make the original image, he sent then user is considered authentic. Else the user is not granted access. The system uses image segmentation based on coordinates. The coordinates of the segmented image allow the system to fragment the image and store it in different parts. Actually, system segments the image into a grid and stores each part accordingly in order. But while logging in the image is shown as broken and in a jumbled order. At this time only the user who provided the image knows what the actual image looks like and he must the parts in horizontal direction from left to right one row at a time according to the order in which parts were arranged in the original image. So, the user is granted access after successful attempt.

There are number of users who purchase products online and make payment through various websites. There are multiple websites who ask user to provide sensitive data such as username, password or credit card details etc. often for malicious reasons. This type of websites is known as phishing website. In order to detect and predict phishing website, we proposed an intelligent, flexible and effective system that is based on using classification Data mining algorithm. We implemented classification algorithm and techniques to extract the phishing data sets criteria to classify their legitimacy. The phishing website can be detected based on some important characteristics like URL and Domain Identity, and security and encryption criteria in the final phishing detection rate. Once user makes transaction through online when he makes payment through the website our system will use data mining algorithm to detect whether the website is phishing website or not. This application can be used by many E-commerce enterprises in order to make the whole transaction process secure. Data mining algorithm used in this system provides better performance as compared to other traditional classifications algorithms. With the help of this system user can also purchase products online without any hesitation. Admin can add phishing website URL or fake website URL into system where system could access and scan the phishing website and by using algorithm, it

will add new suspicious keywords to database. System uses machine learning technique to add new keywords into database.

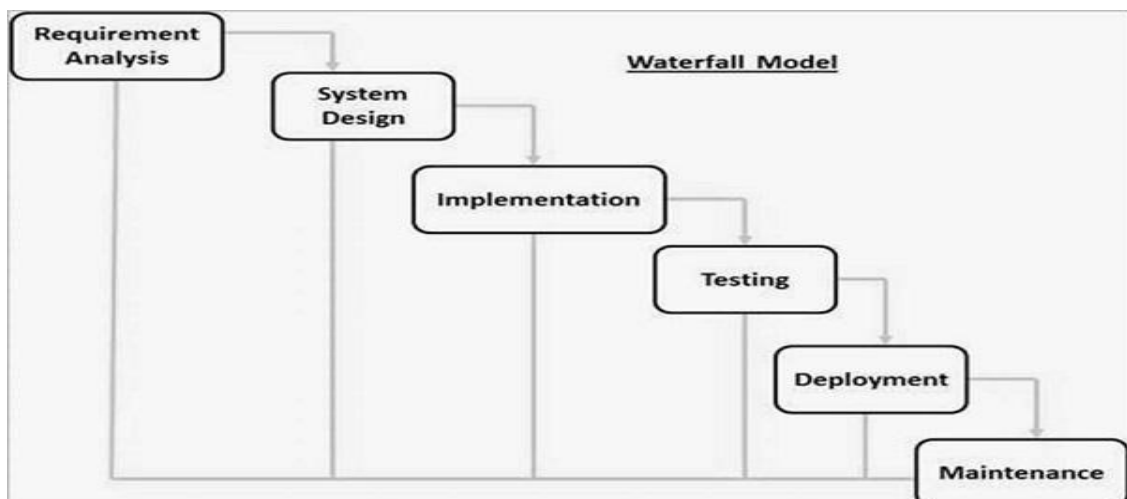
Hardware and software requirements for the installation and smooth functioning of this project could be configured based on the requirements needed by the component of the operating environment that works as front-end system here we suggest minimum configuration for the both hardware and software components. Working off with this software is requirements concrete on system environments.

## 1.1 WATERFALL MODEL:

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially

### Waterfall Model - Design:

In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. Following is the pictorial representation of Waterfall Model:



The sequential phases in Waterfall model are –

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

### **Waterfall Model - Application:**

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are –

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

**The advantages of the waterfall model SDLC Model are as follows:**

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Some of the major advantages of the Waterfall Model are as follows –

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

**The disadvantages of the waterfall model SDLC Model are as follows:**

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.



The major disadvantages of the Waterfall Model are as follows –

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty are high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang" at the very end, which does not allow identifying any technological or business bottleneck or challenges early.

## **CHAPTER II**

### **SYSTEM ANALYSIS**

## **Introduction**

System analysis is the process of collecting and interpreting facts, understanding problems and using the information to suggest improvement on the system. This will help to understand the existing system and determine how computers make its operation more effective. The aim of this analysis is to collect detailed information on the system and the feasibility study of the proposed system.

### **2.1 EXISTING SYSTEM**

- Most systems use textual password for authentication. Textual passwords are easy to recognize. So, another person can easily identify textual passwords. Any person standing beside can easily obtain the passwords. It is more vulnerable to Brute Force attacks.
- Plugins for detecting phishing websites are available in some browsers. But these systems recognize the phishing website only after entering into the website. Users do not have the facility to check whether the URL is phishing before visiting the page.

#### **2.1.1 The Disadvantages of Existing System**

The existing system has the following disadvantages,

- Textual passwords can easily be detected.
- Recognises the phishing website only after entering into the website.

### **2.2 PROPOSED SYSTEM**

- The project allows user to input an image as its password. On receiving the image, the system segments the image into an array of images and stores them accordingly. System segments the image into a grid and stores each part accordingly in order. While logging in the image is shown as broken order. The user is granted access after successful attempt.
- Data mining algorithm used in this system provides better performance as compared to other traditional classifications algorithms. Admin can add phishing website URL or fake website URL into system where system could access and scan the phishing website and by using algorithm, it will add new suspicious keywords to database.

## 2.3. PHISHING WEBSITE FEATURES

### 1. Address bar-based features

If an IP address is used as an alternative of the domain name in the URL, such as “http://125.98.3.123/fake.html”, user can be sure that someone is trying to steal their Personal information. Sometimes, the IP address is even transformed into hexadecimal code as shown in the following link “http://0x58.0xCC.0xCA.0x62/2

paypal.ca/index.html” The Domain Part has an IP Address → Phishing Otherwise → Legitimate

### 1.2 Long URL to Hide the Suspicious Part

Phishers can use long URL to hide the doubtful part in the address bar. For example: `http://federmacedoadv.com.br/3f/aze/ab51e2e319e51502f416dbe46b773a5e/?cmd=home&dispatch=11004d58f5b74f8dc1e7c2e8dd4105e81100d58f5b74f8dc1e7c2e8dd4105e8@phishing.website.html`

### 1.3 Using URL Shortening Services “Tiny URL”

URL shortening is a method on the “World Wide Web” in which a URL may be made considerably smaller in length and still lead to the required webpage. This is accomplished by means of an “HTTP Redirect” on a domain name that is short, which links to the webpage that has a long URL. For example, the URL “http://portal.hud.ac.uk/” can be shortened to “bit.ly/19DXSk4”. Rule: IF Tiny URL → Phishing Otherwise → Legitimate

### 1.4 URL's having “@” Symbol

Using “@” symbol in the URL leads the browser to ignore everything preceding the “@” symbol and the real address often follows the “@” symbol. Rule: IF URL Having @ Symbol → Phishing Otherwise → Legitimate

### 1.5 Redirecting using “//”

The existence of “//” within the URL path means that the user will be redirected to another website. An example of such URL's is:

“http://www.legitimate.com//http://www.phishing.com”. We examine the location where the “//” appears. We find that if the URL starts with “HTTP”, that means the “//”

should appear in the sixth position. However, if the URL employs “HTTPS” then the “/” should appear in seventh position. Rule: IF ThePosition of the Last Occurrence of “/” in the URL > 7 → Phishing Otherwise → Legitimate

## 1.6 Adding Prefix or Suffix Separated by (-) to the Domain

The dash symbol is rarely used in legitimate URLs. Phishers tend to add prefixes or suffixes separated by (-) to the domain name so that users feel that they are dealing with a legitimate webpage. For example, <http://www.Confirme-paypal.com/>.

## 1.7 Sub Domain and Multi Sub Domains

Let us assume we have the following link: <http://www.hud.ac.uk/students/>. A domain name might include the country-code top-level domains (ccTLD), which in our example is “uk”.

The “ac” part is shorthand for “academic”, the combined “ac.uk” is called a second-level domain (SLD) and “hud” is the actual name of the domain. To produce a rule for extracting this feature, we firstly have to omit the (www.) from the URL which is in fact a sub domain

in itself. Then, we have to remove the (ccTLD) if it exists. Finally, we count the remaining dots. If the number of dots is greater than one, then the URL is classified as “Suspicious” since it has one sub domain. However, if the dots are greater than two, it is classified as “Phishing” since it will have multiple sub domains. Otherwise, if the URL has no sub domains, we will assign “Legitimate” to the feature. Rule: IF Dots In Domain Part=1 → Legitimate Dots In Domain Part=2 → Suspicious Otherwise → Phishing

## 1.8 HTTPS

The existence of HTTPS is very important in giving the impression of website legitimacy, but this is clearly not enough. The authors in (Mohammad, Thabtah and McCluskey 2012) (Mohammad, Thabtah and McCluskey 2013) suggest checking the certificate assigned with HTTPS including the extent of the trust certificate issuer, and the certificate age. Certificate Authorities that are consistently listed among the top

trustworthy names include: “GeoTrust, GoDaddy, Network Solutions, Thawte, Comodo, Doster and VeriSign”. Furthermore, by testing out our datasets, we find that the minimum age of a reputable certificate is two years. Rule: IFUse https and Issuer Is Trusted and Age of Certificate 1 Years → Legitimate Using https and Issuer Is Not Trusted → Suspicious Otherwise→ Phishing

## 1.9 Domain Registration Length

Based on the fact that a phishing website lives for a short period of time, we believe that trustworthy domains are regularly paid for several years in advance. Rule: IFDomains Expires on 1 years → Phishing Otherwise→ Legitimate

## 1.10 Favicon

A favicon is a graphic image (icon) associated with a specific webpage. Many existing user agents such as graphical browsers and newsreaders show favicon as a visual reminder of the website identity in the address bar. If the favicon is loaded from a domain other than that shown in the address bar, then the webpage is likely to be considered a Phishing attempt. Rule: IFFavicon Loaded From External Domain→ Phishing Otherwise→ Legitimate

## 1.11 Using Non-Standard Port

This feature is useful in validating if a particular service (e.g. HTTP) is up or down on a specific server. In the aim of controlling intrusions, it is much better to merely open ports that you need. Several firewalls, Proxy and Network Address Translation (NAT) servers will, by default, block all or most of the ports and only open the ones selected. If all ports are open, phishers can run almost any service they want and as a result, user information is threatened. Rule: IFPort is of the Preferred Status→ Phishing Otherwise→ Legitimate

## 1.12 The Existence of “HTTPS” Token in the Domain Part of the URL

The phishers may add the “HTTPS” token to the domain part of a URL in order to trick users. For example, <http://https-www-paypal-it-webapps-mpphome.soft-hair.com/>. Rule: If Using HTTP Token in Domain Part of The URL→ Phishing Otherwise→

Legitimate

## 2. Abnormal based features

### 2.1 Request URL

Request URL examines whether the external objects contained within a webpage such as images, videos and sounds are loaded from another domain. In legitimate webpages, the webpage address and most of objects embedded within the webpage are sharing the same domain.

### 2.2 URL of Anchor

An anchor is an element defined by the `< a >` tag. This feature is treated exactly as “Request URL”. However, for this feature we examine: If the `< a >` tags and the website have different domain names. This is similar to request URL feature. If the anchor does not link to any webpage, e.g.:

- A. `< a href = \" >`
- B. `< a href = \"content\" >`
- C. `< a href = \"skip\" >`
- D. `< a href = \"JavaScript :: void(0)\" >`

### 2.3 Links in `< Meta >`, `< Script >` and `< Link >` tags

Given that our investigation covers all angles likely to be used in the webpage source code, we find that it is common for legitimate websites to use `< Meta >` tags to offer metadata about the HTML document; `< Script >` tags to create a client side script; and `< Link >` tags to retrieve other web resources. It is expected that these tags are linked to the same domain of the webpage.

### 2.4 Server Form Handler (SFH)

SFHs that contain an empty string or “about:blank” are considered doubtful because an action should be taken upon the submitted information. In addition, if the domain name in SFHs is different from the domain name of the webpage, this reveals that the webpage is suspicious because the submitted information is rarely handled by external domains.

Rule: IFSFH is "about: blank" Or Is Empty → Phishing SFH Refers To A Different Domain → Suspicious Otherwise → Legitimate

## 2.5 Submitting Information to Email

Web form allows a user to submit his personal information that is directed to a server for processing. A phisher might redirect the user's information to his personal email. To that end, a server-side script language might be used such as "mail()" function in PHP. One more client-side function that might be used for this purpose is the "mailto:" function. Rule: IF Using "mail()" or "mailto:" Function to Submit User Information → Phishing Otherwise → Legitimate

## 2.6 Abnormal URL

This feature can be extracted from WHOIS database. For a legitimate website, identity is typically part of its URL. Rule: IF The Host Name Is Not Included In URL → Phishing Otherwise → Legitimate

# 3. HTML and JavaScript based features

## 3.1 Website Forwarding

The fine line that distinguishes phishing websites from legitimate ones is how many times a website has been redirected. In our dataset, we find that legitimate websites have been redirected one time max. On the other hand, phishing websites containing this feature have been redirected at least 4 times.

## 3.2 Status Bar Customization

Phishers may use JavaScript to show a fake URL in the status bar to users. To extract this feature, we must dig-out the webpage source code, particularly the "onMouseOver" event, and check if it makes any changes on the status bar.

## 3.3 Disabling Right Click

Phishers use JavaScript to disable the right-click function, so that users cannot view and save the webpage source code. This feature is treated exactly as "Using onMouseOver to hide the Link". Nonetheless, for this feature, we will search for event



“event.button==2” in the webpage source code and check if the right click is disabled.

Rule: IFRight Click Disabled → Phishing Otherwise→Legitimate

### 3.4 Using Pop-up Window

It is unusual to find a legitimate website asking users to submit their personal information through a pop-up window. On the other hand, this feature has been used in some legitimate websites and its main goal is to warn users about fraudulent activities or broadcast a welcome announcement, though no personal information was asked to be filled in through these pop-up windows.

### 3.5 IFrame Redirection

IFrame is an HTML tag used to display an additional webpage into one that is currently shown. Phishers can make use of the “iframe” tag and make it invisible i.e. without frame borders. In this regard, phishers make use of the “frameBorder” attribute which causes the browser to render a visual delineation. Rule: IF Usingiframe→ Phishing Otherwise → Legitimate

## 4. Domain based features

### 4.1 Age of Domain

This feature can be extracted from WHOIS database. Most phishing websites live for a short period of time.

### 4.2 DNS Record

For phishing websites, either the claimed identity is not recognized by the WHOIS database or no records founded for the hostname. If the DNS record is empty or not found then the website is classified as “Phishing”, otherwise it is classified as “Legitimate”. Rule: IFno DNS Record For The Domain → Phishing Otherwise→ Legitimate

### 4.3 Website Traffic

This feature measures the popularity of the website by determining the number of visitors and the number of pages they visit. However, since phishing websites live for a

short period of time, they may not be recognized by the Alexa database (Alexa the Web Information Company., 1996). Rule: IF Website Rank  $\leq$  100,000  $\rightarrow$  Legitimate Website Rank  $>$  100,000  $\rightarrow$  Suspicious Otherwise  $\rightarrow$  Phishing

#### 4.4 PageRank

PageRank is a value ranging from “0” to “1”. PageRank aims to measure how important a webpage is on the Internet. The greater the PageRank value the more important the webpage.

#### 4.5 Google Index

This feature examines whether a website is in Google’s index or not. When a site is indexed by Google, it is displayed on search results. Usually, phishing webpages are merely accessible for a short period and as a result, many phishing webpages may not be found on the Google index.

#### 4.6 Number of Links Pointing to Page

Number of links pointing to the webpage indicates its legitimacy level, even if some links are of the same domain.

#### 4.7 Statistical-Reports Based Feature

Several parties such as PhishTank, and Stop Badware formulate numerous statistical reports on phishing websites at every given period of time; some are monthly and others are quarterly. Rule: IF Host Belongs to Top Phishing IPs or Top Phishing Domains  $\rightarrow$  Phishing Otherwise  $\rightarrow$  Legitimate

## 2.4 Random Forest Algorithm

Random forest is a supervised learning algorithm which is used for both classification as well as regression. But however, it is mainly used for classification problems. As we know that a forest is made up of trees and more trees means more robust forest. Similarly, random forest algorithm creates decision trees on data samples and then gets the prediction from each of them and finally selects the best solution by means of voting. It is an ensemble method which is better than a single decision tree because it reduces the over-fitting by averaging the result.

## Working of Random Forest Algorithm

Step 1 : First, start with the selection of random samples from a given dataset.

Step 2 :Next, this algorithm will construct a decision tree for every sample. Then it will get the prediction result from every decision tree.

Step 3 :In this step, voting will be performed for every predicted result.

Step 4 :At last, select the most voted prediction result as the final prediction result. The below diagram explains the working of the Random Forest algorithm.

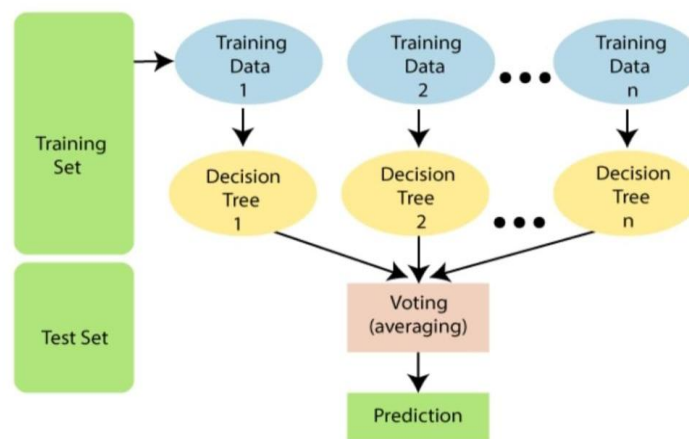


Figure: Working of Random Forest Algorithm

## Features and Advantages

- It is one of the most accurate learning algorithms available. For many data sets, it produces a highly accurate classifier.
- It runs efficiently on large databases.
- It can handle thousands of input variables without variable deletion.
- It gives estimates of what variables are important in the classification.
- It generates an internal unbiased estimate of the generalization error as the forest building progresses.

- It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing.
- It has methods for balancing error in class population unbalanced data sets.
- Generated forests can be saved for future use on other data.
- It offers an experimental method for detecting variable interactions.

## System Testing

Testing is a procedure of executing the program with unequivocal intension of discovering mistakes, assuming any, which makes the program, fall flat. This stage is an essential piece of the product improvement. The objective of testing is to reveal prerequisites, outline or coding blunders in the projects. Therefore, unique levels of testing are utilized in programming frameworks. The testing results are utilized amid upkeep.

This area manages the points of interest in the various classes of the test which should be directed to approve capacities, imperatives and execution. This can be accomplished fundamentally by using the methods for testing, which assumes a crucial part in the improvement of a product.

Here we introduce the performance evaluation of the phishing detection. Dataset consist of 30 features and one label. The 30 features are:

1. Having IP Address
2. URL Length(54)
3. Shortening Service (.com/.in)
4. Having "@" Symbol
5. Double slash redirecting(//)
6. Prefix Suffix
7. Having Sub Domain
8. SSLfinal State
9. Domain registration length

10. Favicon
11. Port
12. HTTPS token
13. Request URL
14. URL of anchor
15. Links in tags
16. SFH (form action)
17. Submitting to email
18. Abnormal URL
19. Redirect
20. On mouseover event
21. RightClick event
22. popup Window
23. Iframe
24. Age of domain
25. DNSRecord
26. Web traffic (rank)
27. Page Rank
28. Google Index
29. Links pointing to page
30. Statistical report

Test cases are exclusively chosen on the premise of the prerequisites or particulars of a program or module of program but the internals of the module of program but the internals of the module or the program are not considered for determination of experiments.

## **2.5 Feasibility Study**

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spent on it. Feasibility study lets the developer foresee the future of the project and the usefulness.

Feasibility study is a test of system proposed regarding its workability, impact on the organization, ability to meet the needs and effective use of resources. Thus, when a new project is proposed, it normally goes through a feasibility study before it is approved for development.

The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as technical, economical and behavioural feasibilities.

The proposed system is theoretically investigated to check the feasibility and found that they are more reliable and efficient in the cases given below. There are three aspects in the feasibility study portion of the preliminary investigation.

✓ Economic feasibility

✓ Technical feasibility

✓ Behavioural feasibility

The proposed system must be evaluated from a technical point of view first, and if technical feasible their impact on the organization must be assessed. If compatible, the operational system can be devised. Then they must be tested for economic feasibility.

### **2.5.1 Economic Feasibility**

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors which affect the development of a new system is the cost it would require. Since the system developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it gives an indication of the system is economically possible for development.

### **2.5.2 Technical Feasibility**

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs, procedures and staff. Having identified an outline system, the investigation must go on suggest the type of equipment, required method developing the system, of running the system once it has been designed. The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology.

Though the technology become obsolete after some period of time, due to the fact that newer version of some software supports older versions, the system still be used. So there are only minimal constraints involved with this project. The system has been developed using PYTHON and ANDROID, along with the database software SQL server, thus we could conclude that the project is technically feasible for development.

### **2.5.3 Behavioural Feasibility**

People are inherently resistant to change and computers have been known to facilitate change. The System is designed in user friendly manner and we need to provide any special training for the persons using this software. The operating system used is Windows 10, which is also user friendly. Since the application is web biased and can easily accessed in a web browser, which is quite familiar to the intended users, it does not have any operational barriers. So no need to provide any special training for using this application software and hence it is behaviourally feasible.

## **2.6 System Specifications**

System Specification deals with the technical aspects the project must meet in minimum to work successfully. This also includes the different aspects the software requirement is determined from.

Hardware and software requirements for the installation and smooth functioning of this project could be configured based on the requirements needed by the component of the operating environment that works as front-end system here we suggest minimum configuration for the both hardware and software components. Working off with this software is requirements concrete on system environments. It includes two phases.

The technical details typically include:

- Software Specification
- Hardware Specification

### **2.6.1 Software Specifications**

The software required for the application depends on the following factors:

- ✓ The flexibility of the software
- ✓ Software contracts
- ✓ Limitation of the software

### **Software Requirement**

This specifies the minimum software requirements for implementing the system. This includes:

- Front End: - Python, Android
- Back End: - My SQL
- Client-side scripting: Java Script
- Server-side scripting: Python
- Software's used: PyCharm/ Sublime, Wamp, SQL yog, Android Studio
- Operating System: Microsoft windows 10

### **2.6.2 Hardware Specifications**

The software required for the application depends on the following factors:

- ✓ Determining size and capacity requirements.
- ✓ Computer evaluation and measurement.
- ✓ Financial factors.
- ✓ Maintenance and support.

### **Hardware Requirement**

- Microprocessor: Any 64-bit processor.
- Clock speed: - 2.13GHz
- Ram: 1 GB and above
- Hard disk: 40 GB and above



- Keyboard: -Standard keyboard
- Mouse: Standard mouse
- Connectivity: - LAN & Wi-Fi

## 2.7 USE CASE DIAGRAMS

### 2.7.1 Identification of Actors

A use cases represents the functionality of an actor. It is defined as a set of actions performed by a system, which yields an observable result. An ellipse containing its name inside the ellipse or below it represents. it. It is placed inside the system boundary and connected to an actor with an association. This shoes how the use cases and the actor interact.

We can identify the actors through a list of questions. The answers to these questions bring out the actors of the system is.

- Admin
- User

Here we need to specify the use cases of each actor.

### 2.7.2 Identification of Use Cases

A use case represents the functionality of an actor. It is defined as a set of actions performed by a system, which yield an observable result. An ellipse containing its name inside the ellipse or below it represents it. It is placed inside the system boundary and connected to an actor with an association. This shows how the use cases and the actor interact.

To find out the use cases, ask the following questions to each of the actors.

- ✓ Which functions does the actor require from the system? What does the actor need to do?
- ✓ Does the actor need to read, create, destroy, modify or store some kind of information in the system?
- ✓ Does the actor have to calculate something? And want to provide information for others?

- ✓ Could the actor's daily work be simplified or made more efficient by adding new functions to the system (typically functions which are currently not automated in the system)?

### **2.7.3 Use cases for the actor-ADMIN**

#### **Login:**

The first step involved is login. The admin can login to the website using username and password.

#### **View registered users:**

Admin can add or remove users.

#### **View complaint:**

Admin can view complaints send by users.

#### **Send reply:**

Admin can view complaints and reply.

#### **View suggestions:**

Admin can view feedbacks send by users.

#### **Load dataset(evaluation):**

Admin can load the datasets.

#### **View prediction report:**

Admin can view the phishing prediction report.

### **2.7.4 Use cases for the actor-USER**

#### **Register:**

The user can register by giving a username and password.

#### **Login:**

The user can login to the website using username and password.

#### **View profile:**

User is able to view the profile.

**Upload files:**

Users can upload files stored in their devices.

**View uploaded files:**

Users can view the files they have uploaded.

**Download files:**

Users can download files.

**Send complaint:**

Users can send complaints to the admin.

**View reply:**

Users can view the reply received from the admin.

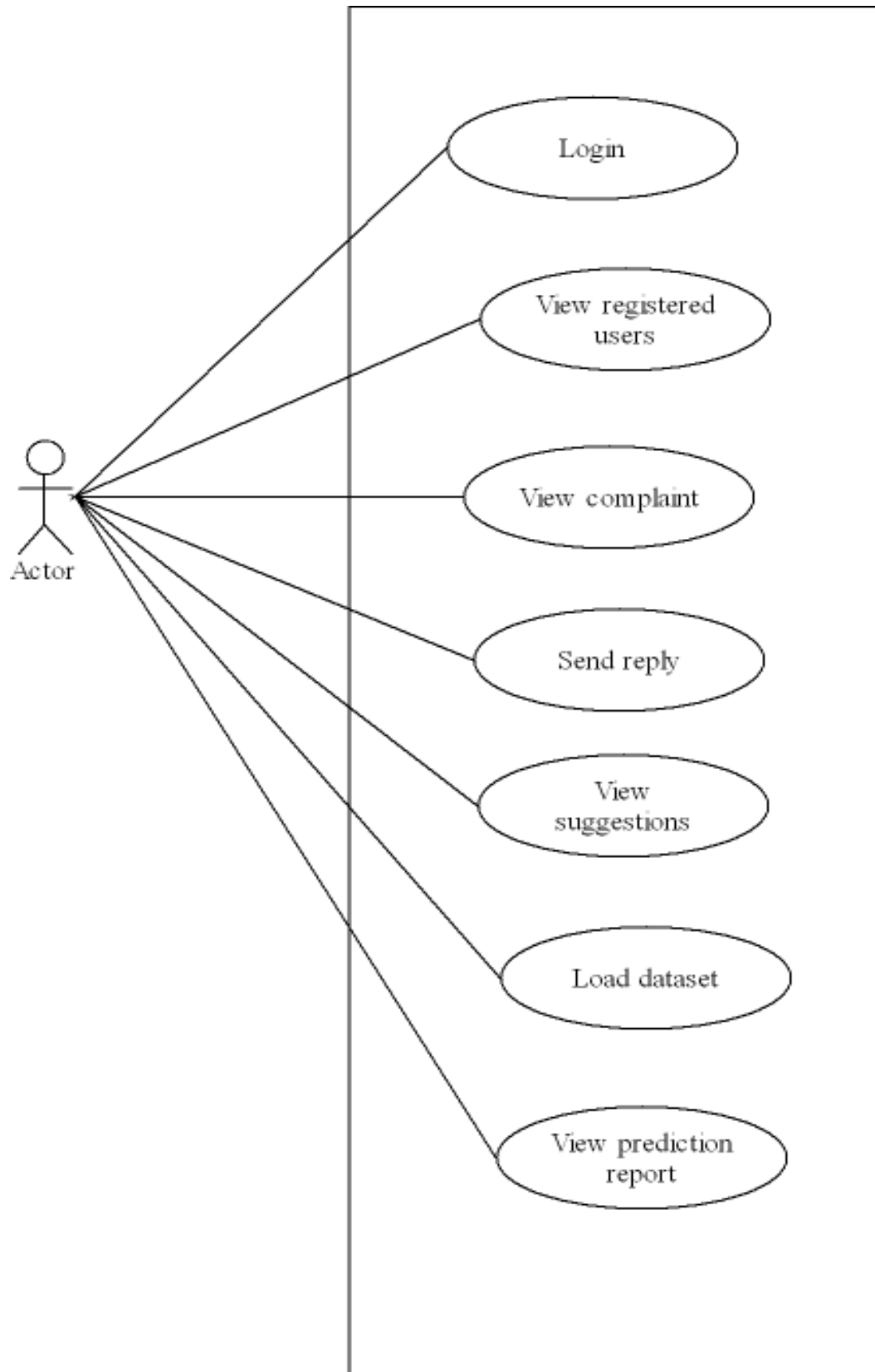
**Send suggestions:**

Users can send their suggestions to the user.

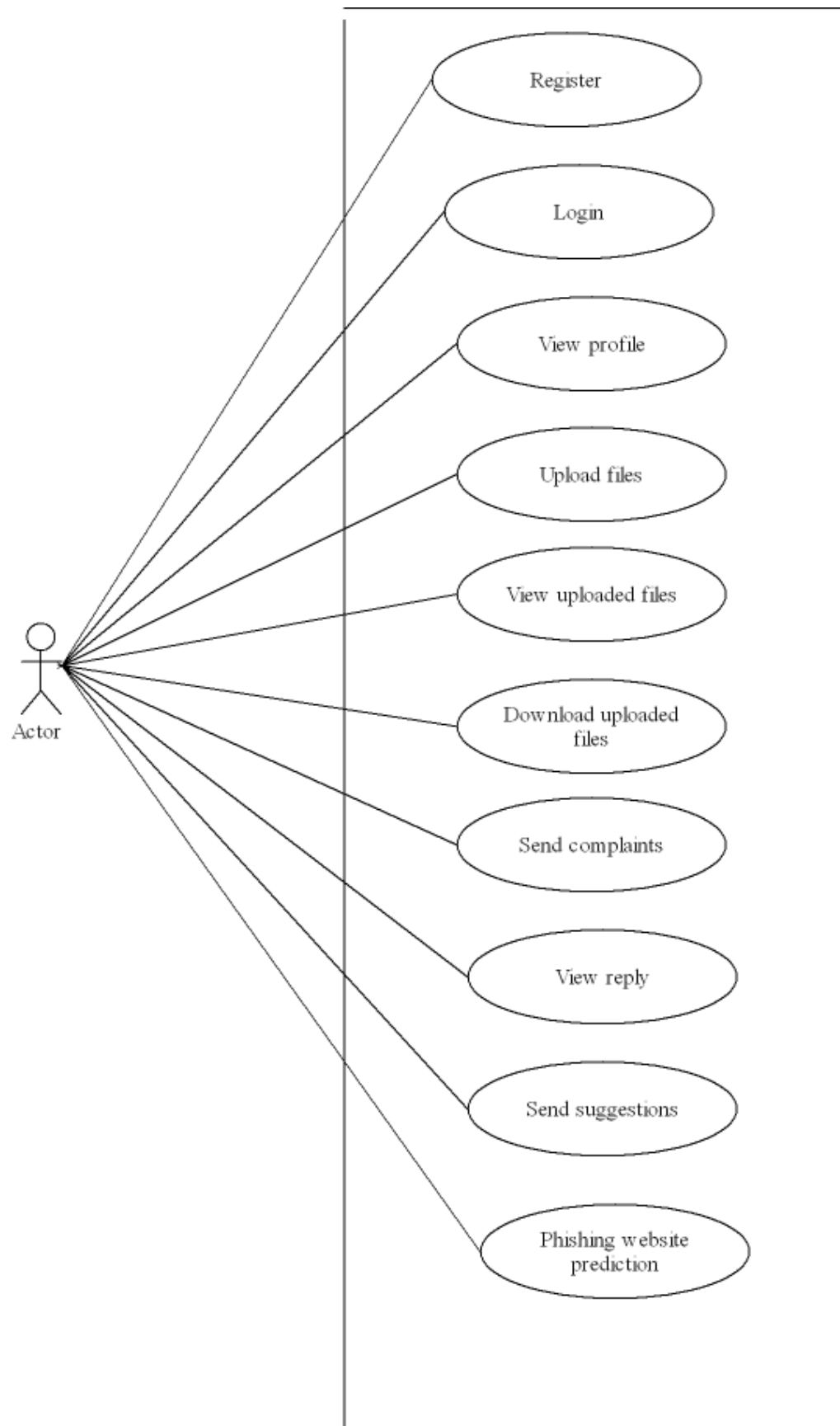
**Phishing website prediction:**

Phishing websites is detected comparing them with 30 features of Random Forest algorithm.

## 2.7.5 USE CASE DIAGRAMS



**USECASE DIAGRAM OF ADMIN**



**USECASE DIAGRAM OF USER**

# **CHAPTER III**

## **SYSTEM DESIGN**

### **3.1 INTRODUCTION:**

System design provides an understanding of the procedural details, necessary implementing the system recommended in the feasibility study. Basically, it is all about the creation of a new system. This is a critical phase since it decides the quality of the system and has a major impact on the testing and implementation phases.

**System design consists of three major steps.**

- Drawing of the expanded system data flow charts to identify all the processing functions required.
- The allocation of the equipment and the software to be used.
- The identification of the test requirements for the system.

### **CHARACTERS OF DESIGN**

- A design should exhibit a hierarchical organization that makes intelligent use of control among components of the software.
- A design should be modular that is, the software should be logical.
- A design should contain distinct and separable representation of data and procedure.
- A design should lead to interface that reduce the complexity of the connections between modules and with the external environment.

### **3.2 Database Design**

A Database is a collection of inter related data stored with minimum redundancy to serve many users quickly and efficiently. In database design data independence, accuracy, privacy and security are given higher priority. Database design is an integrated approach to the file design. This activity deals with the design of the physical data base. All entities and attributes have been identified while creating the database. The database design deals with the grouping of data into number of tables so as to.

- ✓ Reduplication of data.
- ✓ Minimize storage space.
- ✓ Retrieve the data efficiently.

Following are some guidelines for the database design:

- Design a relational schema so that it is easy to explain its meaning. Do not combine attributes from multiple entry and relationship type into a single relation.
- Design the database schema so that no insertion, deletion or modification anomalies are present in the relation.
- As far as possible, avoid placing attributes in the base relation whose values may frequently be null.
- Design relation schema so that they can be joined with equality conditions on attributes that are either primary keys or foreign keys in a way that no spurious tuples are generated.

### 3.3 Table Design

DB design is required to manage large bodies of information. The management of data involves both the definition of the structure of storage of information and provisions of mechanism for the manipulation of information. For developing efficient database certain conditions have to be fulfilled such as:

- Control Redundancy
- Ease of Use
- Data Independence
- Accuracy and Integrity

There are five major steps in design process:

- Identify the table and relationship.
- Identify the data that is needed for each table and relationship.
- Resolve the relationship.
- Verify the design.
- Implement the design

The Database Consist of the following tables given below.



### 3.4 Phishing Website Detection with Graphical Password

*1.Login table*

Column name	Data type	Constraints	Description
login_id	int	primary key	Unique identifier
username	varchar(50)	not null	Name of user
password	varchar(50)	not null	Secret key
usertype	varchar(50)	not null	To specify the type of user

*2.User table*

Column name	Data type	Constraints	Description
user_id	int	primary key	Unique identifier
name	varchar(50)	not null	Name of the user
email	varchar(100)	not null	Email id of the user
phone	varchar(50)	not null	Phone no. of user
image	varchar(100)	not null	Photo of user

*3. Complaint table*

Column name	Data type	Constraints	Description
complaint_id	int	primary key	Unique identifier
user_id	varchar(50)	not null	Identifies the user
date	varchar(50)	not null	Date in which complaint is send
complaint	varchar(250)		Complaint send
reply	varchar(50)	not null	Reply send for the complaint by admin

#### 4. Prediction table

Colum name	Data type	Constraints	Description
prediction_id	int	primary key	Unique identifier
user_id	int	foreign key	Id of the user
date	curdate()		Date of prediction
time	curtime()		Time of prediction
url	varchar(250)	not null	Url entered for detection
prediction	varchar(50)		Result of detection

#### 5. Suggestion table

Colum name	Data type	Constraints	Description
suggestion_id	int	primary key	Unique identifier
user_id	int	foreign key	Id of the user
suggestion	varchar(100)	not null	Suggestion provided by the user
date	curdate()		Date of suggestion send
time	curtime()		Time of suggestion send

#### 6. Uploads table

Colum name	Data type	Constraints	Description
file_id	int	not null	Unique identifier
date	curdate()		Date of file upload
user_id	int	foreign key	Id of the user
title	varchar(50)	Not null	Title for the upload
path	varchar(100)	Not null	Path of the file

### **3.5. Data Flow Diagram**

A graphical representation is used to describe and analyses the movement of data through a system manual or automated including the processes, Storing of data and delays in the system. Data flow diagrams are the central tool and the basis from which other components are developed.

The transformation of data, from input to output through process may be described logically and independently of the physical components associated with the system.

They are termed logical dataflow diagrams, showing the actual implementations and the movement of data between people, departments and

workstations. DFD is one of the most important modelling tools used insystem design. DFD shows the flow of data through different process in the system.

#### **PURPOSE:**

The purpose of the design is to create architecture for the evolving implementation and to establish the common tactical policies that must be used by desperate elements of the system. We begin the design process as soon as we have reasonably completed model of the behavior of the system. It is important to avoid premature designs, wherein develop designs before analysis reaches closer. It is important to avoid delayed designing where in the organization crashes while trying to complete an unachievable analysis model.

Throughout my project, the context flow diagrams, data flow diagrams and flow charts have been extensively used to achieve the successful design of the system. In my opinion, "efficient design of the data flow and context flow diagram helps to design the system successfully without much major flaws within the scheduled time". This is the most complicated part in a project. In the designing process, my project took more than the activities in the software lifecycle. If we design a system efficiently with all the future enhancements the project will never become junk and it will be operational.

The data flow diagrams were first developed by Larry Constantine as a way of expressing system requirements in graphical form. A data flow diagram also known as "bubble chare" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. It functionality

decomposes the requirement specification down to the lowest level. Data Flow Diagram depicts the information flow, the transformation flow and the transformations that are applied as data move from input to output. Thus DFD describes what data flows rather than how they are processed.

Data Flow Diagram is quite effective, especially when the required design is unclear and the user and analyst need a notational language for communication. It is one of the most important tools used during system analysis. It is used to model the system components such as the system process, the data used by the process, any external entities that interact with the system and information flows in the system.

Data Flow Diagrams are made up of a number of symbols, which represents system components. Data flow modelling method uses four kinds of symbols, which are used to represent four kinds of system components.

These are

- Process
- Data stores
- Data flows
- External entity

#### **Process:**

Process shows the work of the system. Each process has one or more data inputs and produce one or more data outputs. Processes are represented by rounded rectangles in Data Flow Diagram. Each process has a unique name and number. This name and number appear inside the rectangle that represents the process in a Data Flow Diagram.

#### **Data Stores:**

A data stores is a repository of data. Processes can enter data, into a store or retrieve the data from the data store. Each data has a unique name.

#### **Data Flows:**

Data flows show the passage of data in the system and are represented by lines joining system components. An arrow indicates the direction of flow and the line is labelled by name of the dataflow.

### **External Entity:**

External entities are outside the system but they either supply input data into the system or use other systems output. They are entities on which the designer has control. They may be an organizations customer or other bodies with which the system interacts. External entities that supply data into the system are sometimes called source. External entities that use the system data are sometimes called sinks. These are represented by rectangles in the Data Flow Diagram.

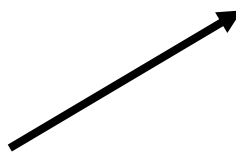
Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

Basic data flow diagram symbols are.....

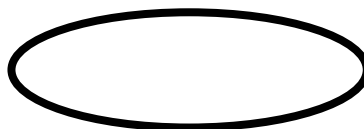
- A Square defines a source (originator) or destination of a system data:



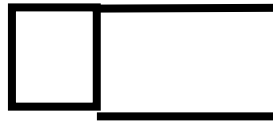
- An Arrow identifies data flow. It is a pipeline through which information flows:



- A Circle represents a process that transforms incoming data flow(s) into outgoing data flow(s):



- An Open Rectangle is a data store:

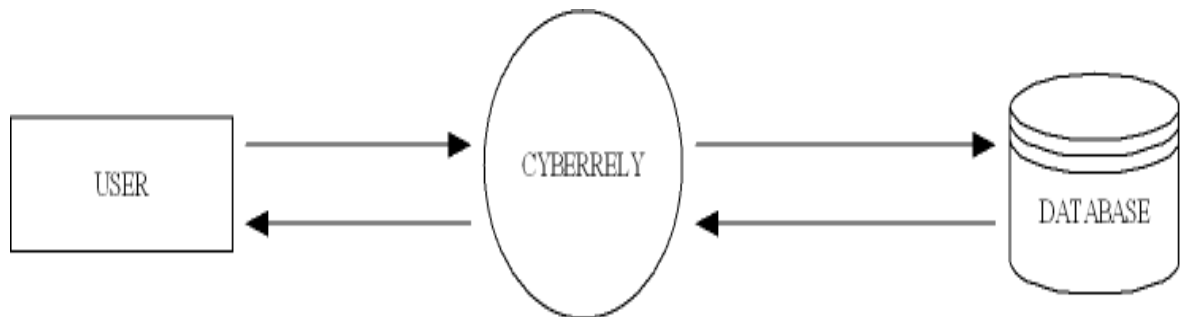


**Four steps are commonly used to construct a DFD:**

- Process should be named and numbered for easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and left to right.
- When a process is exploded in to lower-level details they are numbered.
- The names of data stores, sources and destinations are written in Capital letters.

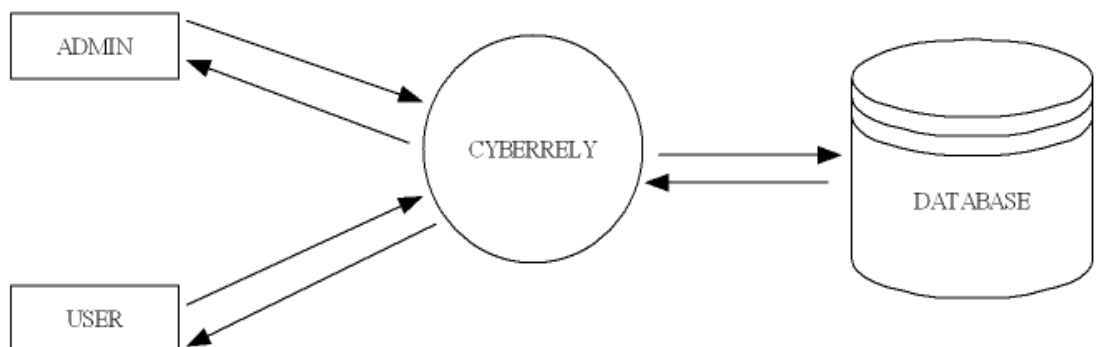
### ***DFD Level-0***

#### LEVEL 0



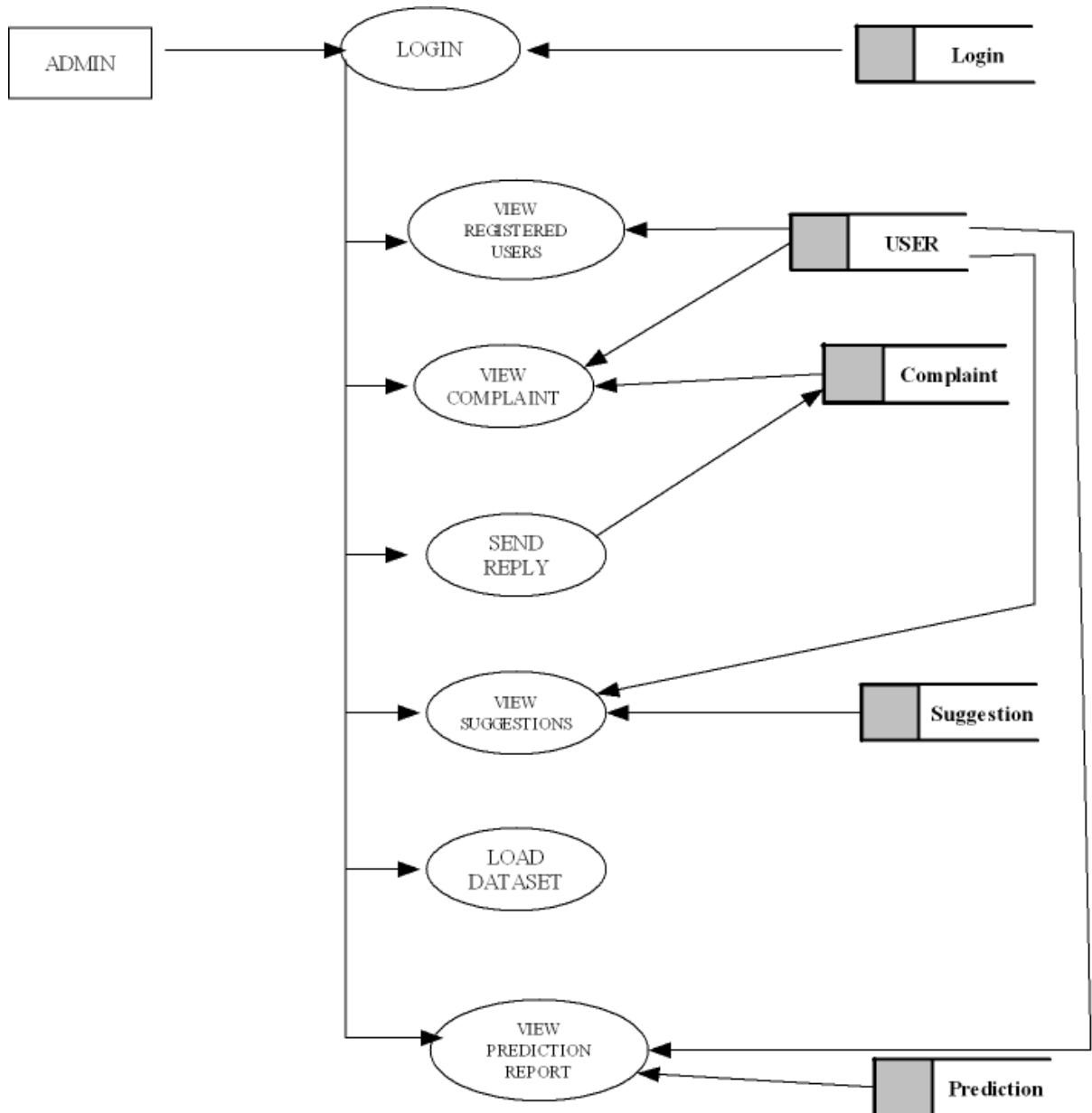
### ***DFD Level-1***

#### LEVEL 1



## DFD Level-2 ADMIN

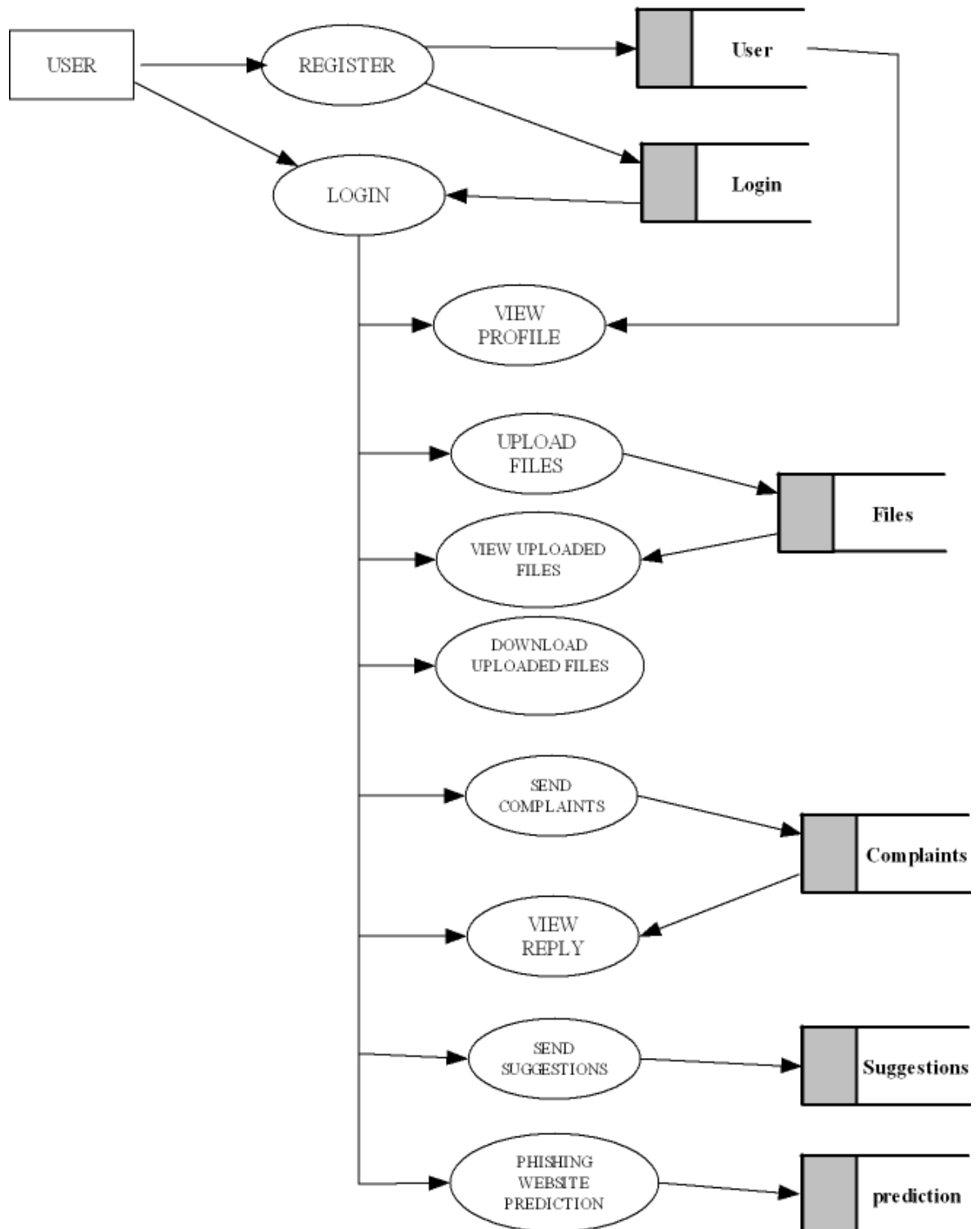
### LEVEL 2 : ADMIN





## DFD Level-2 USER

### LEVEL 2 : USER



### 3.6 ER Diagram

An ER diagram is a diagram that helps to design databases in an efficient way. It is a data model for describing the data or information. It is a visual representation of data that describes how data is related to each other. The main components of ER models are entities, attributes and the relationships that can exist among them.

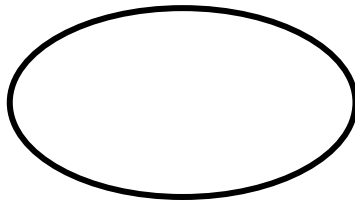
#### Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.



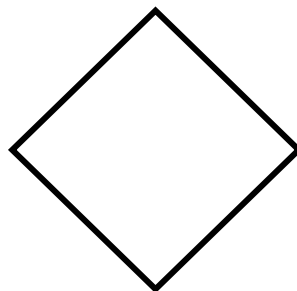
#### Attribute

Attributes are properties of entities. Attributes are represented by means of eclipses. Every eclipse represents one attribute and is directly connected to its entity (rectangle).

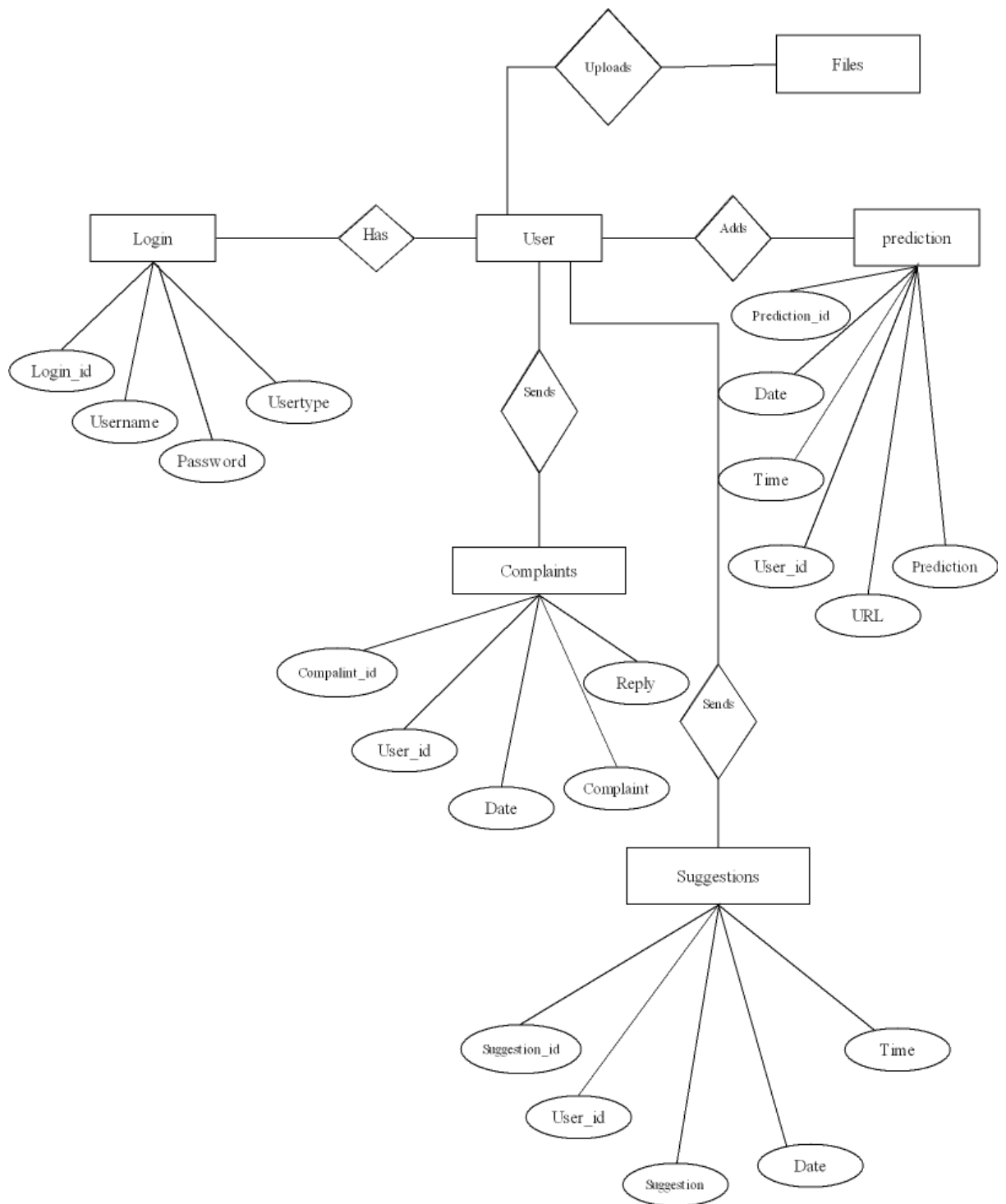


#### Relationship

Relationships are represented by diamond shaped box. Name of the relationship is written in the diamond box. All entities (rectangles), participating in relationship, are connected to it by a line.



## ***ER DIAGRAM***



## **CHAPTER IV**

### **CODING**

## **4.1 INPUT INTERFACE**

Input design is a part of overall system design, which requires very careful attention. If data going into the system is correct, then the processing and output will magnify these errors. Thus the designer has a number of clear objectives in the different stages of input design.

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that input is acceptable to and understand by the user.

## **4.2 OUTPUT INTERFACE**

At the beginning of the output design various types of outputs such as external, internal, operational and interactive and turn around are defined. Then the format, content, location, frequency, volume and sequence of the outputs are specified. The content of the output must be defined in detail. The system analysis has two specific objectives at this stage.

- To interpret and communicate the results of the computer part of a system to the users in a form, which they can understand, and which meets their requirements.
- To communicate the output design specifications to programmers in a way in which it is unambiguous, comprehensive, and capable of being translated into a programming language.

## **4.3 SOFTWARE DESCRIPTION**

### **TECHNOLOGY SPECIFICATION**

The proposed system is developed using HTML, CSS, JavaScript for web and Java(Android Studio) for Android as Front end. MySQL, Flask andPython for Web and Python,MySQL for Android as Back end.

### **FRONT END**

#### **4.3.1 HTML**

HTML is an acronym which stands for Hyper Text Markup Language which is used for creating web pages and web applications. Let's see what is meant by Hypertext Markup Language, and Web page. Hyper Text: Hypertext simply means "Text within

Text." A text has a link within it, is a hypertext. Whenever you click on a link which brings you to a new webpage, you have clicked on a hypertext. Hypertext is a way to link two or more web pages (HTML documents) with each other. Markup language: A markup language is a computer language that is used to apply layout and formatting conventions to a text document. Markup language makes text more interactive and dynamic. It can turn text into images, tables, links, etc. Web Page: A web page is a document which is commonly written in HTML and translated by a web browser. A web page can be identified by entering an URL. A Web page can be of the static or dynamic type. With the help of HTML only, we can create static web pages. Hence, HTML is a markup language which is used for creating attractive web pages with the help of styling, and which looks in a nice format on a web browser. An HTML document is made of many HTML tags and each HTML tag contains different content

### **Features**

- 1) It is a very easy and simple language. It can be easily understood and modified.
- 2) It is very easy to make an effective presentation with HTML because it has a lot of formatting tags.
- 3) It is a markup language, so it provides a flexible way to design web pages along with the text.
- 4) It facilitates programmers to add a link on the web pages (by html anchor tag), so it enhances the interest of browsing of the user.
- 5) It is platform-independent because it can be displayed on any platform like Windows, Linux, and Macintosh, etc.
- 6) It facilitates the programmer to add Graphics, Videos, and Sound to the web pages which makes it more attractive and interactive.
- 7) HTML is a case-insensitive language, which means we can use tags either in lowercase or upper-case.

### **HTML Versions**

Since the time HTML was invented there are lots of HTML versions in market, the brief introduction about the HTML version is given below:

**HTML 1.0:** The first version of HTML was 1.0, which was the barebones version of HTML language, and it was released in 1991.

**HTML 2.0:** This was the next version which was released in 1995, and it was standard language version for website design. HTML 2.0 was able to support extra features such as form-based file upload, form elements such as text box, option button, etc.

**HTML 3.2:** HTML 3.2 version was published by W3C in early 1997. This version was capable of creating tables and providing support for extra options for form elements. It can also support a web page with complex mathematical equations. It became an official standard for any browser till January 1997. Today it is practically supported by most of the browsers.

**HTML 4.01:** HTML 4.01 version was released on December 1999, and it is a very stable version of HTML language. This version is the current official standard, and it provides added support for stylesheets (CSS) and scripting ability for various multimedia elements.

**HTML5 :** HTML5 is the newest version of Hyper Text Markup language. The first draft of this version was announced in January 2008. There are two major organizations one is W3C (World Wide Web Consortium), and another one is WHATWG( Web Hypertext Application Technology Working Group) which are involved in the development of HTML 5 version, and still, it is under development.

### **Description of HTML example**

**<!DOCTYPE>:** It defines the document type or it instruct the browser about the version of HTML

**<html> :** This tag informs the browser that it is an HTML document. Text between html tag describes the web document. It is a container for all other elements of HTML except<!DOCTYPE>

**<head> :** It should be the first element inside the element, which contains the metadata(information about the document). It must be closed before the body tag opens.

**<title>:** As its name suggests it is used to add title of that html page which appears at the top of the browser window. It must be placed inside the head tag and should close immediately (optional).

**<body>** : Text between body tag describes the body content of the page that is visible to the end user . This tag contains the main content of the html document.

**<h1>** : Text between <h1> tag describes the first level heading of the webpage.

**<p>**: Text between <p> tag describes the paragraph of the webpage.

### **4.3.2 CSS (Cascading Style Sheet)**

CSS is used to control the style of a web document in a simple and easy way.CSS is the acronym for "**Cascading Style Sheet**".**Cascading Style Sheets (CSS)** is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

#### **History of CSS**

CSS was first proposed by **HakonWium Lie** on October 10, 1994. At the time, Lie was working with Tim Berners-Lee (father of Html) at CERN. The European Organization for Nuclear Research is known as CERN. HakonWium lie is known as father of CSS.CSS was proposed in 1994 as a web styling language, to solve some of the problems of Html 4. There were other styling languages proposed at this time, such as Style Sheets for Html and JSSS but CSS won.

#### **Why to learn CSS?**

**Cascading Style Sheets**, fondly referred to as **CSS**, is a simple design language intended to simplify the process of making web pages presentable.<sup>23</sup> CSS is a MUST



for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain.

I will list down some of the key advantages of learning CSS:

- **Create Stunning Web site** - CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.
- **Become a web designer** - If you want to start a carrier as a professional web designer, HTML and CSS designing is a must skill.
- **Control web** - CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.
- **Learn other languages** - Once you understand the basic of HTML and CSS then other related technologies like JavaScript, PHP, or Angular are become easier to understand.

### Types of CSS

There are three ways of inserting a style sheet in any Html documents, they are given below:

- Inline style sheet
- Internal style sheet
- External style sheet

Inline CSS is use with any elements of HTML where it is used on page. Here we use inline CSS for paragraph, the example shows how to change the color and the left margin of a paragraph. An internal style sheet should be used when a single document has a unique style. Internal styles sheet is defined in the head section of an HTML page, by using the <style> tag. An external style sheet is ideal when the style is applied to many pages. With an external style sheet, we can change the look of an entire Web site by changing one file.

### CSS Selectors

Selectors are used for select an Html element it is select by name, id, class etc.

1. id selector
2. class selector
3. Element Selector
4. Group Selector
5. Universal Selector

## **Features**

1. A style rule consists of a selector component and a declaration block component.
2. The selector is used to point to the HTML component which you want to get styled.
3. Inside the declaration block, one or more declarations are contained along with semicolons.
4. Every declaration which is put has a CSS property name, a semicolon, and a value. For example, color is the property and the value is red in color. Font size is the property and the 15px is the value.
5. CSS declaration ends with a semicolon and these blocks are surrounded by curly braces.
6. CSS selectors are the ones which are used to find HTML elements which are based on the element name, id, attribute, class and more.
7. One unique element will be selected by the ID of an element.
8. If you wish to select the particular element with a specific id, the # function along with the id attribute should be used.
9. If you wish to select the elements with a specific class, the period character along with the name class should be written.
10. Universal selector: If you are not interested in choosing the elements of a certain type, the universal selector simply matches with the element name.
11. Element selector: These selectors choose the element based on the element name.
12. Descendent selector: When a particular element lies inside another element, then it is called as the descendent selector.
13. ID selector: This selector uses the id of the HTML element so that a specific element could be selected.
14. Class selectors: It selects the element with a specific class attribute.

15. Grouping selectors: It will be a good option to group the selectors so as to minimize the code. Each selector along with a comma should be used to group the selectors.

### 4.3.3 JavaScript

**JavaScript** is a object-based scripting language and it is light weighted. It is first implemented by Netscape (with help from Sun Microsystems). JavaScript was created by Brendan Eich at Netscape in 1995 for the purpose of allowing code in web-pages (performing logical operation on client side). It is not compiled but translated. JavaScript Translator is responsible to translate the JavaScript code which is embedded in browser. Netscape first introduced a JavaScript interpreter in Navigator 2. The interpreter was 25 an extra software component in the browser that was capable of interpreting JavaScript source code inside an HTML document. This means that web page developers no need other software other than a text editor of develop any web page.

#### Why we use JavaScript?

Using HTML, we can only design a web page but you cannot run any logic on web browser like addition of two numbers, check any condition, looping statements (for, while), decision making statement (if-else) at client side. All these are not possible using HTML so for perform all these tasks at client side you need to use JavaScript

#### History

JavaScript is an object-based scripting language and it is light weighted. It is first implemented by Netscape (with help from Sun Microsystems). JavaScript was created by Brendan Eich at Netscape in 1995 for the purpose of allowing code in web-pages (performing logical operation on client side). Using HTML we can only design a web page but you cannot run any logic on web browser like addition of two numbers, check any condition, looping statements (for, while), decision making statement (if-else) etc. All these are not possible using HTML so to perform all these tasks, we use JavaScript. Using HTML, we can only design a web page if we want to run any programs like C programming, we use JavaScript. Suppose we want to print sum of two numbers then we use JavaScript for coding.

## Features

- JavaScript is an object-based scripting language.
- Giving the user more control over the browser.
- It Handling dates and time.
- It Detecting the user's browser and OS
- It is light weighted.
- JavaScript is a scripting language and it is not java.
- JavaScript is interpreter-based scripting language.
- JavaScript is case sensitive.
- JavaScript is object-based language as it provides predefined objects.
- Every statement in JavaScript must be terminated with semicolon (;).
- Most of the JavaScript control statements syntax is same as syntax of control statements in C language. An important part of JavaScript is the ability to create new functions within scripts. Declare a function in JavaScript using function keyword.

## Android

**Android** is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language. Android applications are written in the Java programming language. The Android SDK tools compile the code—along with any data and resource files—into an Android package, an archive file with an .apk suffix. All the code in a single .apk file is considered to be one application and is the file that Android-powered devices use to install the application. Application components are the essential building blocks of an Android application. Each component is a different point through which the system can enter your application. Not all components are actual entry points for the user and some depend on each other, but each one

exists as its own entity and plays a specific role—each one is a unique building block that helps define application's overall behaviour.

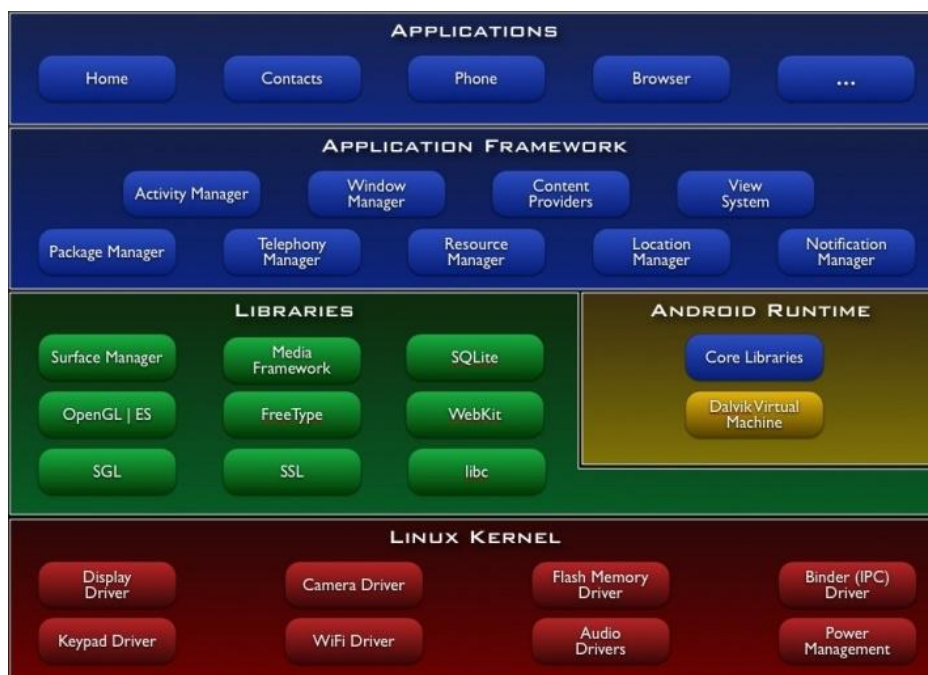
## Features

- Application framework enabling reuse and replacement of components
- Dalvik virtual machine optimized for mobile devices

- Integrated browser based on the open source Web Kit engine
- Optimized graphics powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
- Media support for common audio, video, and still image formats (MPEG4,H.264, MP3, AAC, AMR, JPG, PNG, GIF)
- GSM Telephony (hardware dependent)
- Bluetooth, EDGE, 3G, and Wi-Fi (hardware dependent)
- Camera, GPS, compass, and accelerometer (hardware dependent)
- Rich development environment including a device emulator, tools for debugging, memory and performance profiling, and a plug-in for the Eclipse IDE.

## ANDROID ARCHITECTURE

The following diagram shows the major components of the Android operating system. Each section is described in more detail below.



## APPLICATION FRAMEWORK

By providing an open development platform, Android offers developers the ability to build extremely rich and innovative applications. Developers are free to take advantage of the device hardware, access location information, run background services, set alarms, add notifications to the status bar, and much, much more.

Developers have full access to the same framework APIs used by the core applications. The application architecture is designed to simplify the reuse of components; any application can publish its capabilities and any other application may then make use of those capabilities (subject to security constraints enforced by the framework). This same mechanism allows components to be replaced by the user.

Underlying all applications is a set of services and systems, including:

A rich and extensible set of the views that can be used to build an application, including lists, grids, text boxes, buttons, and even an embeddable web browser

Content Providers that enable applications to access data from other applications (such as Contacts), or to share their own data

A Resource Manager, providing access to non-code resources such as localized strings, graphics, and layout files

A Notification Manager that enables all applications to display custom alerts in the status bar

An Activity Manager that manages the lifecycle of applications and provides a common navigation back stack.

## **Libraries**

Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below:

- System C library - a BSD-derived implementation of the standard C system library (libc), tuned for embedded Linux-based devices
- Media Libraries - based on Packet Video's Open CORE; the libraries support playback and recording of many popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG
- Surface Manager - manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications
- LibWebCore - a modern web browser engine which powers both the Android browser and an embeddable web view
- SGL - the underlying 2D graphics engine

- 3D libraries - an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer
- Free Type - bitmap and vector font rendering<sup>24</sup>

## **Android Runtime**

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language. Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple VMs efficiently. The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been transformed into the dex format by the included "dx" tool. The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management. Linux Kernel Android relies on Linux version 2.6 for core system services such as security, memory management, process management, network stack, and driver model. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack.

## **Activity Lifecycle**

Activities in the system are managed as an activity stack. When a new activity is started, it is placed on the top of the stack and becomes the running activity -- the previous activity always remains below it in the stack, and will not come to the foreground again until the new activity exits.

An activity has essentially four states:

- If an activity in the foreground of the screen (at the top of the stack), it is active or running.
- If an activity has lost focus but is still visible (that is, a new non-full sized or transparent activity has focus on top of your activity), it is paused. A paused activity is completely alive (it maintains all state and member information and remains attached to the window manager), but can be killed by the system in extreme low memory situations.

- If an activity is completely obscured by another activity, it is stopped. It still retains all state and member information, however, it is no longer visible to the user so its window is hidden and it will often be killed by the system when memory is needed elsewhere.
- If an activity is paused or stopped, the system can drop the activity from memory by either asking it to finish, or simply killing its process. When it is displayed again to the user, it must be completely restarted and restored to its previous state.

#### 4.3.4 Android Studio

**Android Studio** is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (ADT) as primary IDE for native Android application development. Android Studio supports all the same programming languages of IntelliJ, and PyCharm and Android Studio 3.0 supports Java 7 language features and a subset of Java 8 language features that vary by platform version. Features like Gradle-based build support, Android-specific refactoring and quick fixes, a rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations, Android Virtual Device (Emulator) to run and debug apps in the Android studio, etc. are provided in the current stable version.

#### 4.3.5 JAVA

The first version of Java began in 1991 and was written in 18 months at Sun micro system. In fact, it wasn't even called Java in those days it was Oak, and it was used internally at sun.

Java had adopted a model that made it perfect for the Internet, the byte code model. It is implemented as the Java virtual Machine (JVM), which is the application that actually runs the java program. 26 When JVM is installed on a computer, it can run java programs. Java programs, before, don't need to be self-sufficient, and they don't have to include all the machine –level code that actually runs on the computer. In this



way, our Java program can be very small, because all the machine-level code to run our program is already on the target computer and doesn't have to be downloaded.

When it executes a program, the JVM can strictly monitor what goes on, which makes it great for Internet Applications.

## **JAVA FEATURES**

The inventors of java wanted to design a language, which could offer solutions to some of the problems encountered in modern programming. They wanted the language to be reliable, portable and distributed but also simple, compact and interactive. Sun Microsystems officially describes java with the following attributes.

- Compiled and Interpreted
- Platform-Independent and Portable
- Object-Oriented
- Robust and Secure
- Distributed
- Familiar, Simple and Small
- Multithreaded and Interactive
- High Performance
- Dynamic and Extensible

### **Compiled and Interpreted**

Usually, a computer language is either compiled or interpreted. Java combines both these approaches thus making java a two-stage system first java compiler translate source code in to byte code instructions. Byte-codes are not machine code that can be directly executed by the machine that is running the java program. Platform Independent and Portable.

The most significant contribution java over other languages is its portability. Java programs can be easily moved from one computer system to another, anywhere at any time. Changes and Upgrades in 27 operating systems, processors and system resources will not force any changes in java programs. This is the reason why java has become a popular language for programming on internet, which interconnects different kinds of systems worldwide. Java ensures portability in two ways. First, java compiler

generates byte code instructions that can implemented on any machine. Secondly, the sizes of the primitive data types are machine independent.

### **Object-Oriented**

Java is a true Object-Oriented Language. Almost everything in Java is an Object. All program code and data reside within objects and classes. Java comes with an extensive set of classes arranged in packages that we can use in our programs by inheritance. The object model in java is simple and easy to extend.

### **Robust and Secure**

Java is a robust language. It provides many safe guards to ensure reliable code. It has strict compile time checking for data types. It is designed as garbage collected language. Java also incorporates with the concept of exception handling.

### **Distributed**

Java is designed as a distributed language for creating application on network. It has the ability to share both data & Program.

### **Multithreaded and interactive**

Multithreaded means handling multiple tasks simultaneously java supports multithreaded programs. This means that we not wait for the application to finish one task before beginning another.

### **High performance**

Java performance is impressive for an interpreted language mainly due to the use of intermediate byte code. Java architecture is also designed to reduce overheads during runtime.

## **BACK END**

### **4.3.6 Python**

Python is an object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well

as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

### **4.3.7 MySQL**

MySQL software is Open Source.

Open Source means that it is possible for anyone to use and modify. Anybody can download the MySQL software from the Internet and use it without paying anything. The MySQL database server is very fast, reliable, and easy to use. It was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Though under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet

- An object-oriented interface
- Support for prepared statements
- Support for multiple statements
- Support for transactions

- Enhanced debugging support
- Embedded server support

### 4.3.8 PyCharm

**PyCharm** is an Integrated Development Environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains (formerly known as IntelliJ). It provides code analysis, a graphical debugger, an integrated unit tester, integration with Version Control Systems (VCSes), and supports web development with Django as well as data science with Anaconda

#### FEATURES

- Coding assistance and analysis, with code completion, syntax and error highlighting, linear integration, and quick fixes
- Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages

Python refactoring: includes rename, extract method, introduce variable, introduce constant, pull up, push down and others

- Integrated Python debugger
- Integrated unit testing, with line-by-line code coverage
- Version control integration: unified user interface for Mercurial, Git, Subversion, Perforce and CVS with change lists and merge

### 4.3.9 Flask

**Flask** is a micro web framework written in Python. It is classified as micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

**CHAPTER V**  
**CODING PAGES**

## 5.1 Admin Page

```
import base64
from email.mime.text import MIMEText

from flask import Flask, render_template, request, redirect, jsonify
import random

from future.backports.email.mime.multipart import MIMEMultipart

import phishing_detection
from DBConnection import Db

app = Flask(__name__)
static_path=r"C:\Users\Dell\PycharmProjects\cyberrely\static\"

@app.route('/logout')
def logout():
    return redirect("/")

@app.route('/')
def login():
    return render_template("login.html")

@app.route('/forgot')
def forgot():
    return render_template("forgotpassword.html")

@app.route("/forgot_post",methods=['post'])
def forgot_password():
    email=request.form['textfield']
    db=Db()
    res=db.selectOne("select password from login where username='"+str(email)+"' ")
    if res is not None:

import smtplib

        s = smtplib.SMTP(host='smtp.gmail.com', port=587)
        s.starttls()

        s.login("jeswinjames312002@gmail.com", "dqzzelcsjwlekofr")
        msg = MIMEMultipart() # create a message....."
        msg['From'] = "jeswinjames312002@gmail.com"
        msg['To'] = email
        msg['Subject'] = "Your Password for cyberrely Website"
        body = "Your pattern for selecting image location is:- " + str(res['password'])
        msg.attach(MIMEText(body, 'plain'))
        s.send_message(msg)
    return '<script>alert("password send");window.location="/"</script>'
    else:
    return '<script>alert("invalid email");window.location="/"</script>'
```

```

@app.route("/login_post", methods=['post'])
def login_post():
    username=request.form['textfield']
    password=request.form['textfield2']
    db=Db()
    res=db.selectOne("select * from login WHERE username='"+username+"' and
password='"+password+"'")
    if res is not None:
        if res['usertype']=='admin':
            return redirect('/admin_home')
        else:
            return '<script>alert("Unauthorised user");window.location="/"</script>'
        else:
            return '<script>alert("Invalid details");window.location="/"</script>'

@app.route("/admin_home")
def admin_home():
    return render_template("index.html")
    # return render_template("home.html")

@app.route("/send_reply/<cid>")
def send_reply(cid):
    return render_template("send_reply.html", cid=cid)

@app.route("/send_reply_post", methods=['post'])
def send_reply_post():
    reply=request.form['textarea']
    cid=request.form['hid']
    db=Db()
    db.update("update complaint set reply='"+reply+"' where
complaint_id='"+str(cid)+"'")
    return '<script>alert("reply send");window.location="/view_complaint"</script>'

@app.route("/view_complaint")
def view_complaint():
    db=Db()
    res=db.select("select * from complaint,USER where complaint.user_id=user.user_id")
    return render_template("view_complaint.html", data=res)

@app.route("/view_prediction")
def view_prediction():
    db=Db()
    res=db.select("select * from prediction,user where prediction.user_id=user.user_id
order by prediction_id")
    return render_template("view_prediction.html", data=res)

@app.route("/delete_prediction/<pid>")

```

```

def delete_prediction(pid):
    db=Db()
    db.delete("delete from prediction where prediction_id='"+pid+"'")
    return redirect("/view_prediction")

@app.route("/view_registered_user")
def view_registered_user():
    db=Db()
    res=db.select("select * from user")
    return render_template("view_registered_user.html", data=res)

@app.route("/view_suggestion")
def view_suggestion():
    db=Db()
    res=db.select("select * from suggestion, user where suggestion.user_id=user.user_id")
    return render_template("view_suggestion.html", data=res)

```

##### ANDROID

```

@app.route("/and_login", methods=['post'])
def and_login():
    username=request.form['u']
    password=request.form['p']
    db=Db()
    res=db.selectOne("select * from login WHERE username='"+username+"' and
password='"+password+"'")
    if res is not None:
        if res['usertype']=='user':
            return jsonify(status="ok", lid=res['login_id'])
        else:
            return jsonify(status="no")
        else:
            return jsonify(status="no")

@app.route("/and_register", methods=['post'])
def and_register():
    name=request.form['name']
    email=request.form['email']
    phone=request.form['phone']
    pswd=request.form['pswd']
    image=request.form['image']
    import time
    timestr = time.strftime("%Y%m%d-%H%M%S")
    print(timestr)
    a = base64.b64decode(image)
    fh = open(r"C:\Users\Dell\PycharmProjects\cyberrely\static\user_img\\" + timestr +
".jpg", "wb")
    path = "/static/user_img/" + timestr + ".jpg"
    fh.write(a)
    fh.close()
    db=Db()

```



```

    lid=db.insert("insert into login(username, password, usertype)
values('"+email+"','"+pswd+"','user')")
    db.insert("insert into user(user_id,name,email,phone,image)
values('"+str(lid)+"','"+name+"','"+email+"','"+phone+"','"+path+"')")
return jsonify(status="ok")

```

```

@app.route("/and_viewreply",methods=['post'])
def and_viewreply():
    lid=request.form['lid']
    db=Db()
    res=db.select("select * from complaint where user_id='"+lid+"'")
return jsonify(status="ok", data=res)

```

```

@app.route("/and_deletecomplaint", methods=['post'])
def and_deletecomplaint():
    complaint_id=request.form['complaint_id']
    db=Db()
    res=db.delete("delete from complaint where complaint_id='"+complaint_id+"'")
return jsonify(status="ok")

```

```

@app.route("/and_sendcomplaint",methods=['post'])
def and_sendcomplaint():
    lid=request.form['lid']
    complaint=request.form['complaint']
    db=Db()
    db.insert("insert into complaint values(null,'"+lid+"',
curdate(), '"+complaint+"','pending')")
return jsonify(status="ok")

```

```

@app.route("/and_fileuploads",methods=['post'])
def and_fileuploads():
    lid=request.form['lid']
    title=request.form['title']
    image=request.form['image']
import time
    timestr = time.strftime("%Y%m%d-%H%M%S")
    print(timestr)
    a = base64.b64decode(image)
    fh = open(r"C:\Users\Dell\PycharmProjects\cyberrely\static\user_img\\" + timestr +
".jpg", "wb")
    path = "/static/user_img/" + timestr + ".jpg"
    fh.write(a)
    fh.close()
    db=Db()
    db.insert("insert into uploads values(null, curdate(), '"+lid+"','"+title+"','"+path+"')")
return jsonify(status="ok")

```

```

@app.route("/and_viewuploads", methods=['post'])
def and_viewuploads():
    lid=request.form['lid']

```

```

    db=Db()
    res=db.select("select * from uploads where user_id='"+lid+"'")
for i in res:
        i['fname']=i['path'].split("/")[-1]
return jsonify(status="ok", data=res)

```

```

@app.route("/and_deleteuploads", methods=['post'])
def and_deleteuploads():
    file_id=request.form['file_id']
    db=Db()
    res=db.delete("delete from uploads where file_id='"+file_id+"'")
    return jsonify(status="ok")

```

```

@app.route("/and_sendsuggestion", methods=['post'])
def and_sendsuggestion():
    lid=request.form['lid']
    suggestion=request.form['suggestion']
    db=Db()
    db.insert("insert into suggestion
values(null,'" +lid+"','"+suggestion+"',curdate(),curtime())")
    return jsonify(status="ok")

```

```

@app.route("/url_predict", methods=['post'])
def url_predict():
    url=request.form['url']
    lid=request.form['lid']
    result = phishing_detection.getResult(url)
    print(url)
    if "http://" not in url or "https://" not in url:
        url= "http://" +url
    print(result)
    db=Db()
    db.insert("insert into prediction
values(null,curdate(),curtime(),'"+str(lid)+"','"+url+"','"+result+"')")
    return jsonify(status="ok", result=result)

```

```

@app.route("/load_graphical", methods=['post'])
def load_graphical():
    lst=[{'path' : '/static/graphical_password/aa.png'},{'path' :
'/static/graphical_password/bb.png'}, {'path' :
'/static/graphical_password/cc.png'},{'path' :
'/static/graphical_password/dd.png'},{'path' :
'/static/graphical_password/ee.png'},{'path' : '/static/graphical_password/ff.png'},{'path'
: '/static/graphical_password/gg.png'},{'path' :
'/static/graphical_password/hh.png'},{'path' : '/static/graphical_password/ii.png'}]
    return jsonify(status="ok", data=lst)

```

```

@app.route("/load_graphical1", methods=['post'])
def load_graphical1():
    idx = request.form['idx']

```

```

    lst =
['/static/graphical_password/aa.png', '/static/graphical_password/bb.png', '/static/graphical
_password/cc.png', '/static/graphical_password/dd.png', '/static/graphical_password/ee.png
', '/static/graphical_password/ff.png', '/static/graphical_password/gg.png', '/static/graphical
_password/hh.png', '/static/graphical_password/ii.png']
    img = lst[int(idx)]
    print("SSS ", img)
    filename = img.split("/")[-1]
    fname = filename.split(".")[0]
    ext = filename.split(".")[1]
    res=[]
for i in range(1, 10):
        res.append({'path' : "/static/cut_images/" + fname + "_" + str(i) + "." + ext})
return jsonify(status="ok", data=res)

@app.route("/and_viewprofile", methods=['post'])
def and_viewprofile():
    lid = request.form['lid']
    db = Db()
    res = db.selectOne("select * from user where user_id='"+lid+"'")
return jsonify(status="ok", data=res)

@app.route("/and_forgotpassword", methods=['post'])
def and_forgotpassword():

    e=request.form['email']
    # pwd=random.randint(0000,9999)
    db = Db()
    res=db.selectOne("select password from login where username='"+str(e)+"' ")
if res is not None:

import smtplib

        s = smtplib.SMTP(host='smtp.gmail.com', port=587)
        s.starttls()
        s.login("jeswinjames312002@gmail.com", "dqzzelcsjwlekofr")
        msg = MIMEMultipart() # create a message....."
        msg['From'] = "jeswinjames312002@gmail.com"
        msg['To'] = e
        msg['Subject'] = "Your Password for cyberrely Website"
        body = "Your pattern for selecting image location is:- " + str(res['password'])
        msg.attach(MIMEText(body, 'plain'))
        s.send_message(msg)
return jsonify(status="ok")
    else:
return jsonify(status="none")

if __name__ == '__main__':
    app.run(host="0.0.0.0", port=8000)

```

## 5.2 User page

### Android Manifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.dell.cyberrely">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"
    />
    <uses-permission
    android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <application
    android:allowBackup="true"
    android:icon="@drawable/appicon2"
    android:label="@string/app_name"
    android:roundIcon="@drawable/appicon2"
    android:supportsRtl="true"
    android:theme="@style/AppTheme"
    android:usesCleartextTraffic="true">
        <uses-library
        android:name="org.apache.http.legacy"
        android:required="false" />
        <activity android:name=".ip_set">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".login" />
        <activity android:name=".register" />
        <activity android:name=".file_upload" />
        <activity android:name=".send_complaint" />
        <activity android:name=".send_suggestion" />
        <activity android:name=".predict_url" />
        <activity android:name=".view_uploads" />
        <activity android:name=".reg2" />
        <activity android:name=".view_reply" />
        <activity android:name=".log1" />
        <activity android:name=".log2" />
        <activity
        android:name=".home_page"
        android:label="@string/title_activity_home_page"
        android:theme="@style/AppTheme.NoActionBar" />
        <activity android:name=".reg1" />
        <activity android:name=".url_prediction" />
        <activity android:name=".secure_site" />
        <activity android:name=".not_secure" />
        <activity android:name=".view_profile" />
        <activity android:name=".forgot_password" />
        <activity android:name=".view_uploads_more"></activity>
    </application>
```

# **CHAPTER VI**

## **SYSTEM TESTING**

## 6.1 TESTING AND EVALUATION

Testing is a process of executing a program with the intent of finding an error. Software testing is a critical element of software quality assurance and represents the ultimate review of specifications, design and coding. Testing includes verifications of the basic logic of each program and verification that the entire system works properly. Testing demonstrates that software functions appear to be working according to specification. In addition, data collected as testing is conducted provides a good indication of software quality as a whole. The debugging process is the most unpredictable part of the testing process. Testing begins at the module level and works towards the integration of the entire computer-based system. Testing and debugging are different activities, but any testing includes debugging. A strategy for software testing must accommodate low level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system function, against customer requirements. No testing is complete without verification and validation. The goals of verification and validation activities are to assess and improve the quality of work products generated during the development and modification of the software. There are two types of verification: life cycle verification and formal verification. Life cycle verification is the process of determining the degree to which the products of the given phase of the development cycle fulfil the specification established during the prior process. Formal verification is the rigorous mathematical demonstration that source code conforms to its specifications. Validation is a process of evaluating software at the end of the software development process to determine conformance with the requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. The primary objectives, when we test software are the following:

- Testing is a process of executing with the intent of finding an error.
- A good test case is one that has a high probability of finding an undiscovered error.
- A successful test is one uncovers undiscovered errors.

Thus, testing plays a very critical role in determining the reliability and efficiency of the software and hence is a very important stage in software development.

Tests are to be conducted on the software to evaluate its performance under a number of conditions. Ideally, it should so at the level of each module and also when all of them are integrated to form the completed system. Software testing is done at different levels.

## **6.2 TESTING STRATEGIES**

A strategy for software testing integrates software test case design method in to a well-planned series of steps that result in the successful construction of the software. The strategy provides a road map that describes the step to be conducted as part of testing, when these steps are planned and then undertaken, and how much effort, time and resources will be required. Therefore any testing strategy must incorporate test planning, test case, design, test execution and resultant data collection and evaluation. A software testing strategy should be flexible enough to promote a customized testing approach. At the same time, it must be rigid enough to promote reasonable planning and management tracking as the project progresses.

**The general characteristics of software testing strategies are:**

- Testing begins at the component level and works “outward” toward the integration of the entire computer system.
- Different testing techniques are appropriate at different points in time.

A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

A strategy must provide guidance for the practitioner and set of milestones for the manager. Because the step on the test strategy occurs at a time when deadline pressure begins to rise, progress must be measurable and problem must surface as early as possible.

The software team’s approach to testing is defining a plan that describes an overall strategy and a procedure that defines specific testing steps and tests that will be conducted. In the proposed system, if the administrator makes any attempt to login to the application without entering his password, then the system will not allow the user to login to the application.

## **6.3 TESTING TECHNIQUES**

The various testing techniques are given below:

### **6.3.1 WHITE BOX TESTING**

White-box testing is also called as glass-box testing, is a test case design method that goes to the control structure of the procedural design to derive

test cases. Using white box testing methods, the software engineer can derive test cases that,

- ✓ Guarantee that all independent paths within a module have been exercised at Least once.
- ✓ Exercise all logical decision on their true and false sides.
- ✓ Execute all loops at their boundaries and within their operational sides.
- ✓ Exercise internal data structure to ensure their validity.

White box testing was successfully conducted on our system. All independent paths within a module have been executed at least once and all logical decisions have been exercised on their true and false sides.

### **6.3.2 BLACK BOX TESTING**

Black-box testing is also called as behavioural testing, focuses on the functional requirement of the software. It is a complimentary approach that is likely uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories.

- ✓ Incorrect or missing functions.
- ✓ Interface errors.
- ✓ Error on data structures or external database access.
- ✓ Behaviour or performance errors.
- ✓ Initialization and termination errors.

Black box testing was successfully conducted on your system. The system was divided into a number of modules and testing was conducted on each module. We have tested the system for incorrect or missing functions, interface and performance errors.



### **6.3.3 UNIT TESTING**

Unit testing comprises the set of tests performed by an individual programmer prior to the integration of the system. Testing removes residual bugs and improves the reliability of the system.

Testing allows the developer to find out the design faults if any, and enable correction if needed. Exhaustive unit testing has to be carried out to ensure the validity of the data. In order to successfully test the entire package, unit testing is carried out. Each module was tested as when it was developed. Thus, it proved easier to conduct minute testing operation and correct them then and there.

### **6.3.4 INTEGRATION TESTING**

Bottom-up integration is the traditional strategy used to integrate the component of a software system into a functional whole. Bottom-up integration consists of unit testing, followed by subsystem testing and followed by testing of the entire system. Unit testing has the goal of discovering errors in the individual parts of the system.

Parts are tested in isolation from one another in an artificial environment known as “Test Harness”, which consists of driver programs and data necessary to exercise the modules. Unit testing should be as exhaustive as possible to ensure that each representative case handled by each module has been tested. Unit testing is eased by a system structure that is composed of small loosely coupled modules.

Both control and data interfaces must be tested. Large software system may require several levels of subsystem testing. Lower-level subsystems are successively combined to form higher level subsystems. In most software systems, exhaustive testing of subsystem capabilities is not feasible due to the combination complexity of the module interfaces. Therefore, test cases must be carefully chosen to exercise the interfaces in the desired manner.

### **6.3.5 ACCEPTANCE TESTING**

Acceptance testing involves planning and execution of functional tests, performance tests and stress tests in order to demonstrate that the implemented system satisfies its requirements. It is not unusual for two sets of acceptance test to be run, those developed by the quality group and those developed by the customer.

In addition to the functional and performance tests, stress tests are performed to determine the limitation of the system. For example, a compiler might be tested to determine the effect of the symbol table overflow, or real-time system might be tested to determine the effect of simultaneous arrival of numerous high priority interrupts.

#### **6.3.6 OUTPUT TESTING**

Output testing of the proposed system is important since no system could be useful if it does not produce the required output.

The output format on the screen is found to be correct, as the format was designed in the system design phase according to the user needs. For the hard copy also the output comes out as the specified requirements by the user. Hence output testing doesn't result in any correction on the system.

## **CHAPTER VI I**

### **SYSTEMIMPLEMENTATIONANDDEPLOYMENT**

Implementation is the process of deploying the new system in the operational environment. Proper implementation is essential to provide a reliable system to meet the organizational requirement. There are four types of implementation methods. They are Direct Changeover, Phased Implementation, Parallel Run and Pilot Approach. The most commonly used implementation methods are Pilot Approach and Parallel Run.

The system which is developed as a web application hence the other functions as normal application, as usual some web development technologies are used in the implementation of the project. The language I selected to program this software is Python. The reason I selected Python is that is a simple and powerful language that especially developed to create web application.

Technologies used in the development of the software are:

- ✓ Programming language – Python
- ✓ DBMS – MySQL server
- ✓ Development tool – PyCharm
- ✓ Development platform – Windows 10

The front end is HTML, and CSS and back end is MySQL Server and Python. The system developed on PyCharm in Windows 10 operating system.

**CHAPTER VIII**  
**CONCLUSION**

The “Phishing website detection with graphical password” has been developed for all given conditions and it is found working effectively under the all the circumstances that may arise in the real environment. The software has been developed to reducing the operator work.

This system is user friendly and is well efficient to make easy interaction with the users of system. The systems provides more security to the users. The system is done with an insight into the necessary modifications that may be required in the future. Hence the system can be maintained successfully without much work.

## **8.1 FUTURE ENHANCEMENT**

As a future venture, it is suggested to make some changes to provide more services and to provide information at right time in right manner.

The current system is securing the image files. As a future enhancement it can also be used to secure pdf format or any other type of files. The graphical password can be made more complex and secure by adding more images and displaying it in jumbled order. All the functions have been done carefully and successfully in the software, and if any development is necessary in future, it can be done without affecting the design by adding additional modules to the system.

**CHAPTER IX**  
**REFERENCE**

- YouTube
- Books:
  - Database System Concept: Martin F.Korth
  - Programming with Java: A Primer, E. Balaguruswami
  - Apress Software Engineering: Rajib Mall
- Websites:
  - W3Schools.com
  - Python.org
  - Stackoverflow.com
  - Flaticon.com

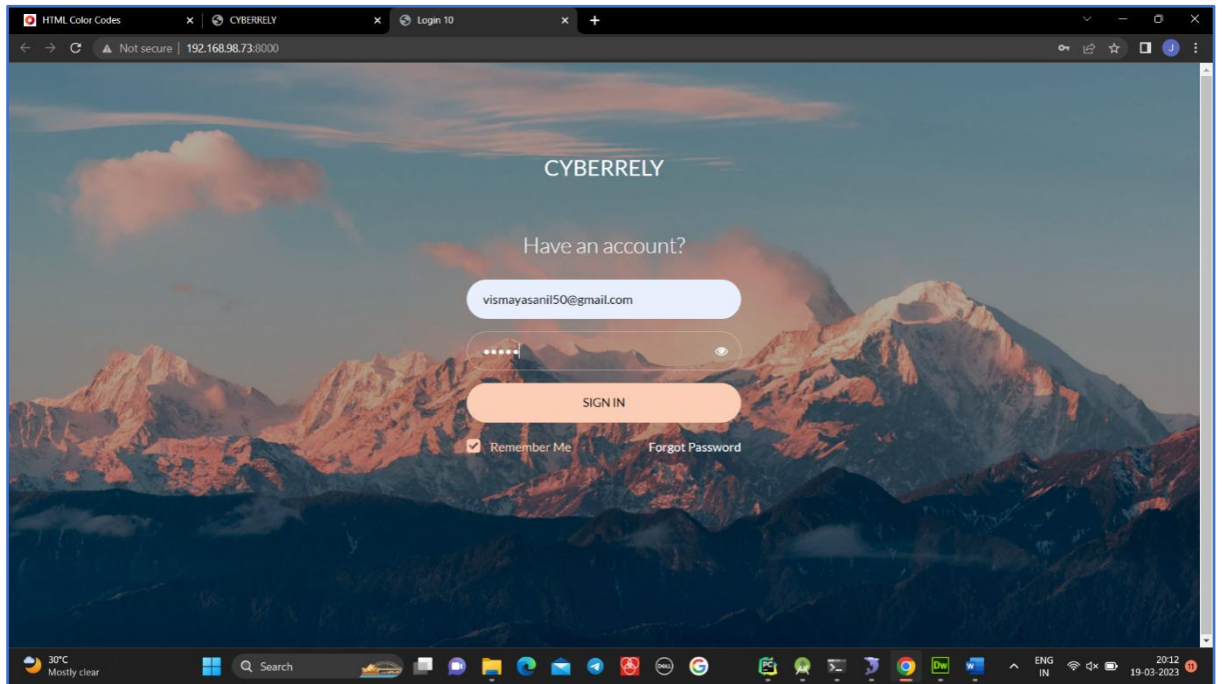


**CHAPTER X**  
**APPENDIX**

## 10.1 Screenshots

### WEBPAGES

#### 10.1.1 AdminLogin:



#### 10.1.2 Admin Homepage



### 10.1.3 Users

SL.NO	NAME	EMAIL	PHONE	IMAGE
1	whajbsn	dhjendn	2778193788	
2	snskndn	shehbdn	2582787810	
3	vismaya	178e993	5789299289	
4	gshssb	hsjsnx	3629927896	
5	xhjb	fgchh	3456776543	
6	vismaya	hsjsjn	2683939086	

### 10.1.4 Complaints

SL NO	USER INFO	DATE	COMPLAINT	REPLY
1	gshssb	2023-01-12	shdhdcdb	ok
2	xhjb	2023-02-28	jjdnxnxn	<a href="#">Reply</a>
3	xhjb	2023-02-28	suhdbzjsnzn	request is being processed
4	xhjb	2023-03-14	sjxbdnccndx	ok done
5	xhjb	2023-03-16	sjxbxb	<a href="#">Reply</a>
6	xhjb	2023-03-18	shzbjsjnn	<a href="#">Reply</a>
7	xhjb	2023-03-18	djwzbwjns	<a href="#">Reply</a>
8	xhjb	2023-03-18	uwjsbdbbd	<a href="#">Reply</a>
9	xhjb	2023-03-18	iwjsvdbddn	<a href="#">Reply</a>

### 10.1.5 Suggestions

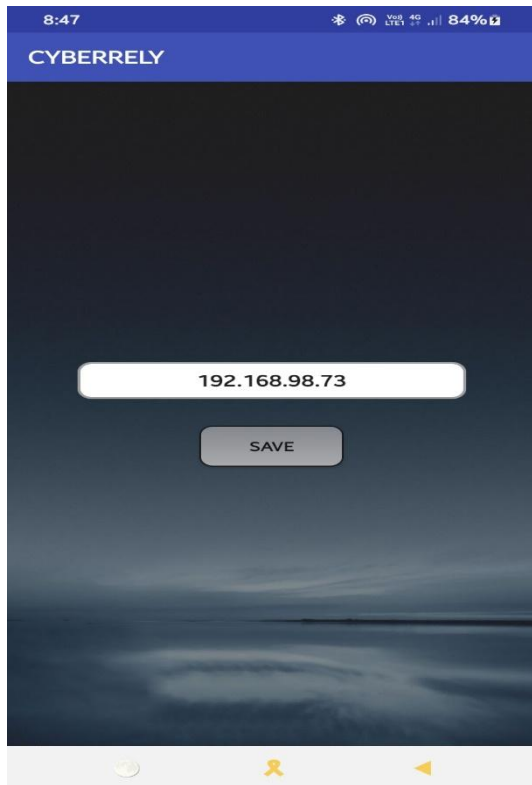
SL NO	USER INFO	SUGGESTION	DATE	TIME
1	xhjb	duehxksnxnn	2023-02-28	11:06:55
2	xhjb	fyyvubibboknbi	2023-03-14	15:20:56
3	xhjb	sjsjdbdbjdks	2023-03-16	21:32:15
4	xhjb	sjsjbsbwjz snannssnjsznjsnssjsksnsn	2023-03-19	18:38:38
5	xhjb	hhebsgwsbwbzjsnwbnsbs	2023-03-19	18:38:45

### 10.1.6 Prediction

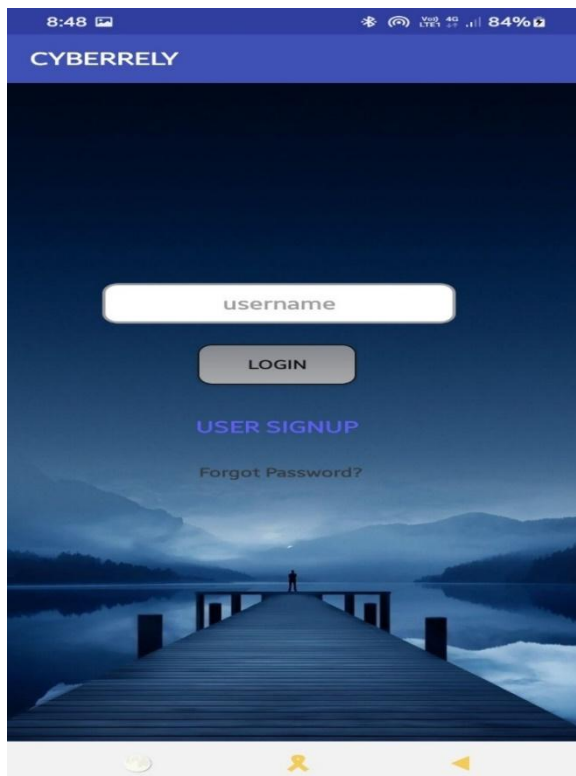
SL.NO	DATE	TIME	USER INFO	URL	
1	2023-03-14	14:16:44	xhjb	http://www.facebook.com Prediction : Legitimate URL	Delete
2	2023-03-14	14:26:36	xhjb	192.168.42.73:5000/sjzbbz@.com Prediction : Phishing URL	Delete
3	2023-03-16	21:36:26	xhjb	http://www.youtube.com Prediction : Legitimate URL	Delete
4	2023-03-17	17:05:43	xhjb	http://www.flipkart.com Prediction : Legitimate URL	Delete
5	2023-03-18	11:57:42	xhjb	http://www.amazon.com Prediction : Legitimate URL	Delete
6	2023-03-18	11:58:27	xhjb	http://www.donbosco.ac.in Prediction : Legitimate URL	Delete
7	2023-03-18	12:00:12	xhjb	http://www.google.com Prediction : Legitimate URL	Delete
8	2023-03-18	12:09:08	xhjb	http://www.facebook.com Prediction : Legitimate URL	Delete
9	2023-03-18	21:01:43	xhjb	http://www.facebook.com Prediction : Legitimate URL	Delete
10	2023-03-19	12:23:43	xhjb	http://192.168.42.73:5000/djjsbzb@.com Prediction : Phishing URL	Delete

## ANDROID

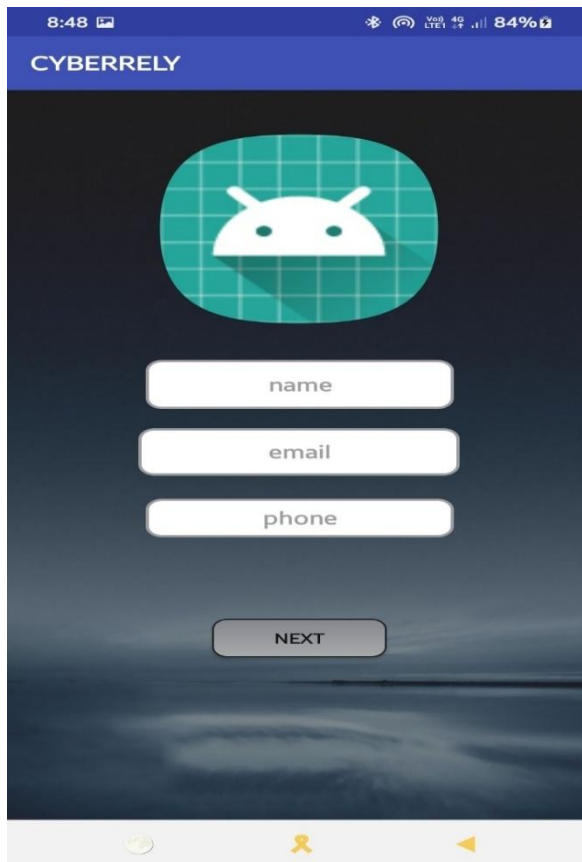
### *User IP Connection*



### *User Login*

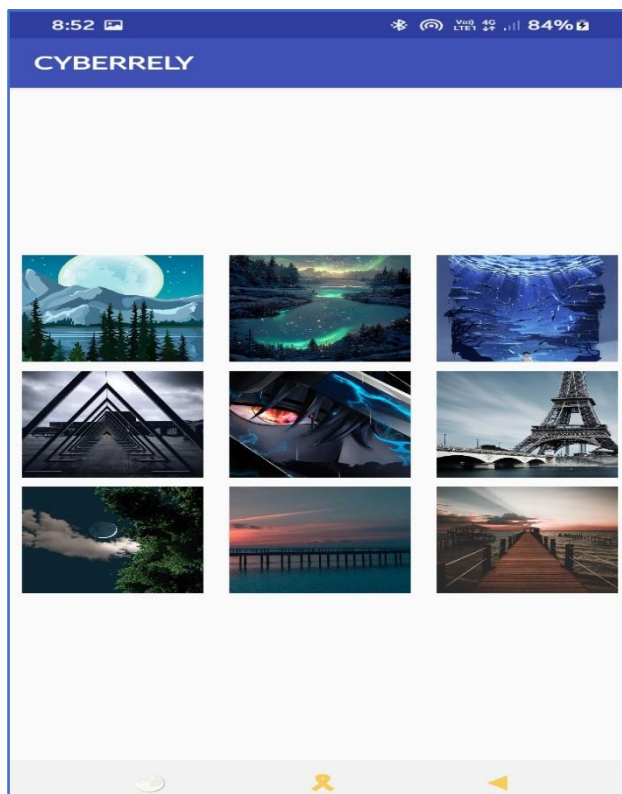


## *User Registration*

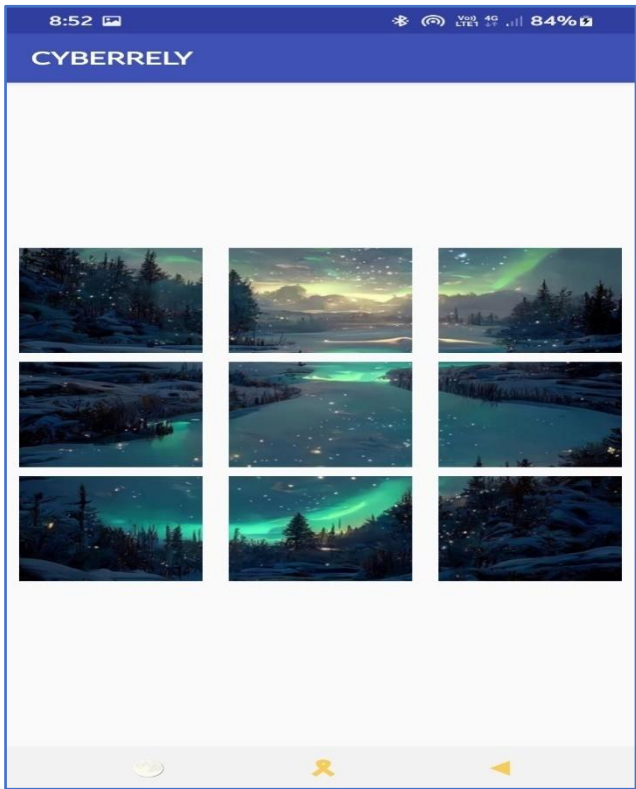


The screenshot shows the 'User Registration' screen of the CYBERRELY app. At the top, the status bar displays the time 8:48, signal strength, LTE1 4G, and 84% battery. The app's header is a blue bar with the text 'CYBERRELY'. Below the header is a large green circle containing a white Android robot icon. Underneath the icon are three white input fields labeled 'name', 'email', and 'phone'. A grey button labeled 'NEXT' is positioned below the input fields. The background of the screen is a dark, atmospheric image of a body of water at night. At the bottom, there is a navigation bar with three icons: a home icon, a person icon, and a back arrow.

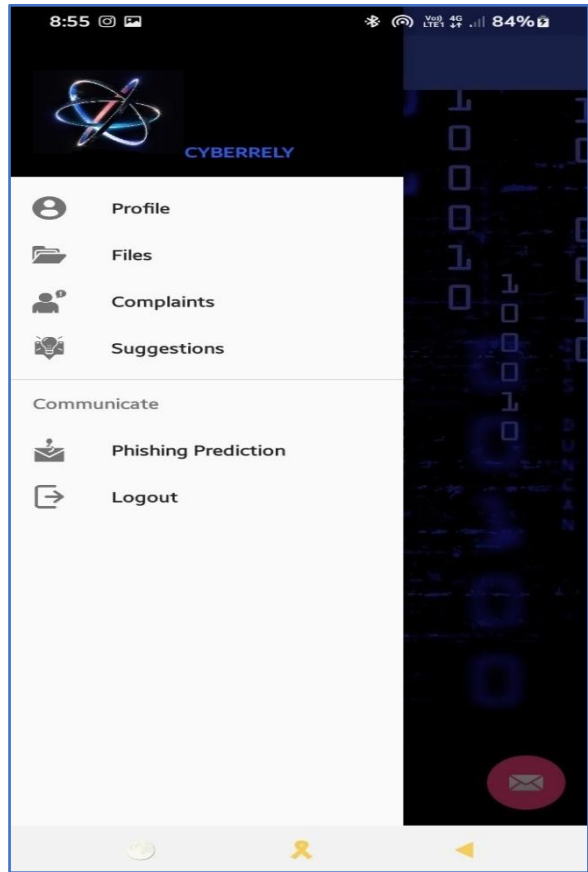
## *Graphical Password-1*



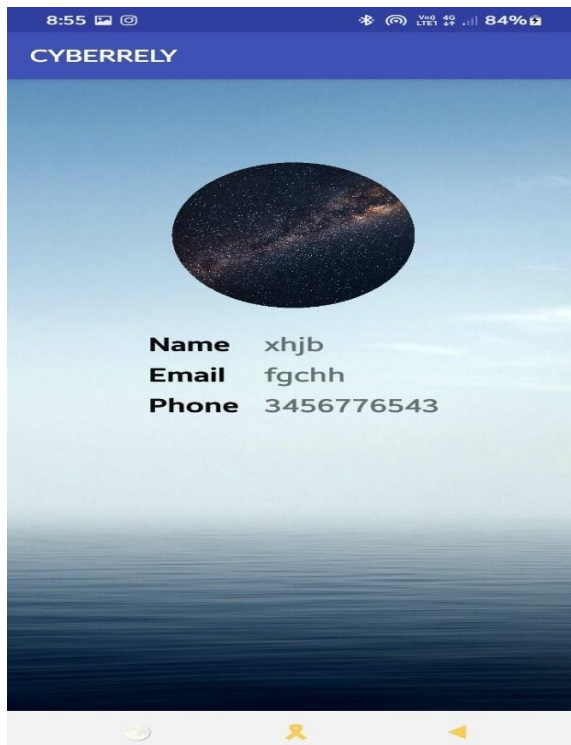
Graphical password-2



Home Page



### View Profile

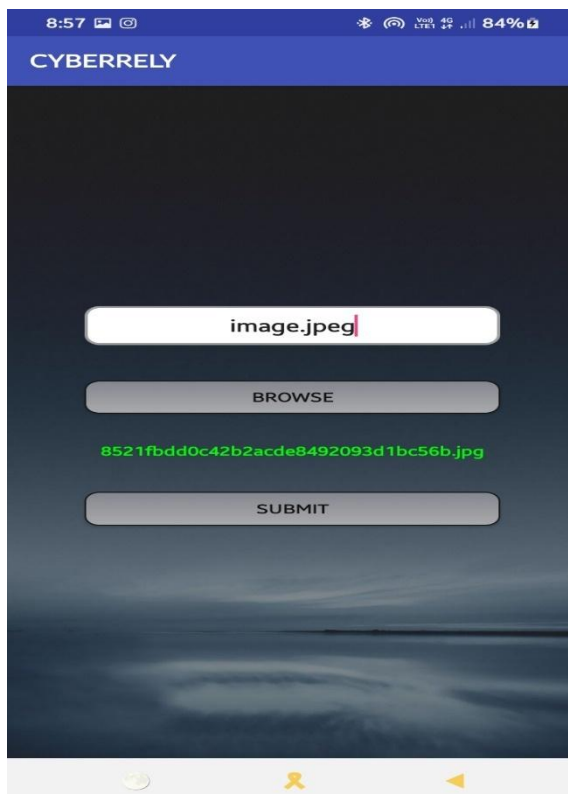


### View Uploads





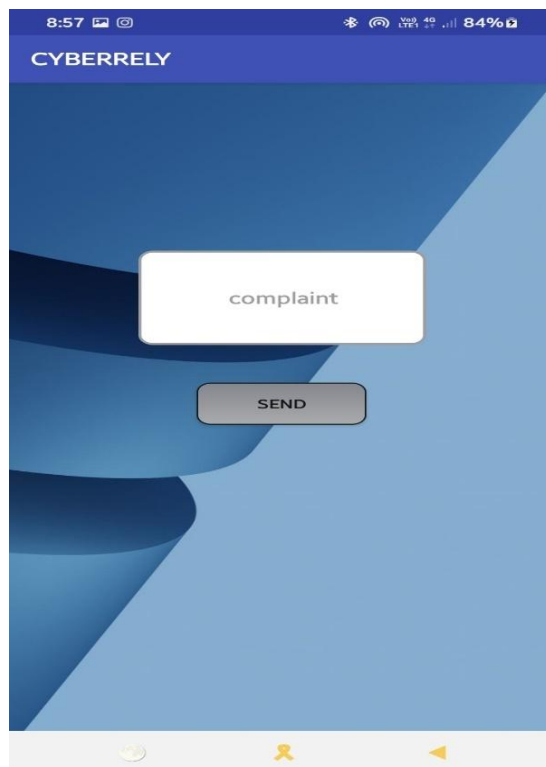
## Upload files



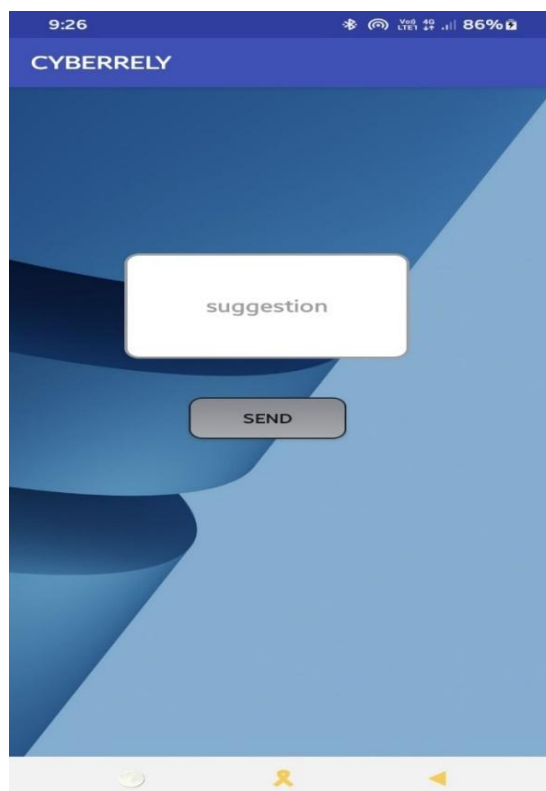
## View Complaints



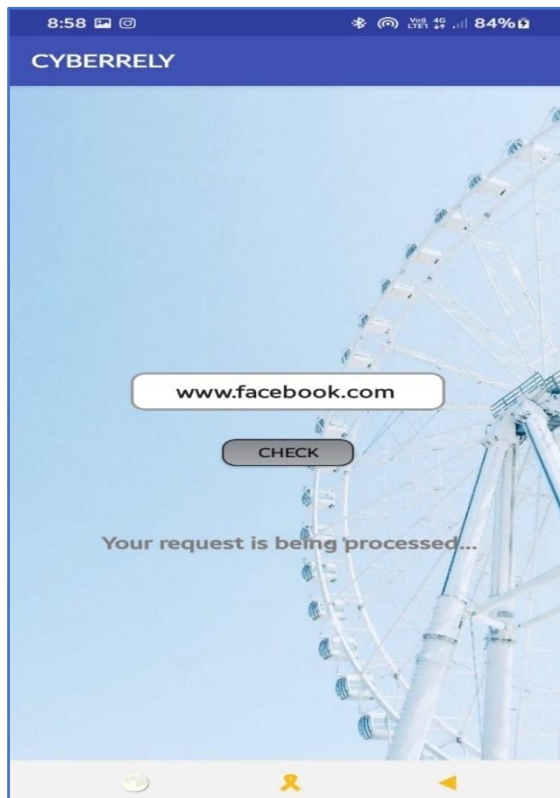
### *Send complaint*



### *Send Suggestion*



## *Phishing Prediction*



## *Phishing Prediction*

