V-CARE DISEASE PREDICTION SYSTEM

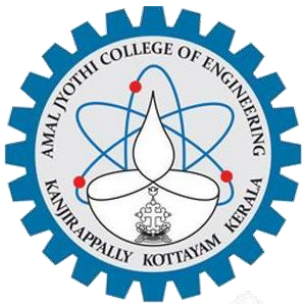
Project Report Submitted By

ASHISH WILSON

Reg. No.: AJC20MCA-2028

In Partial fulfillment for the Award of the Degree Of

REGULAR MASTER OF COMPUTER APPLICATIONS (MCA)
APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY



AMAL JYOTHI COLLEGE OF ENGINEERING
KANJIRAPPALLY

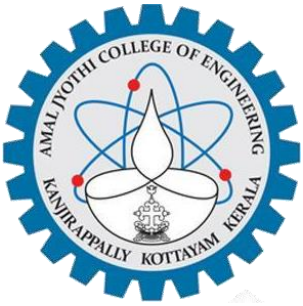
[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC with 'A' grade. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

2021-2022

DEPARTMENT OF COMPUTER APPLICATIONS

AMAL JYOTHI COLLEGE OF ENGINEERING

KANJIRAPPALLY



CERTIFICATE

This is to certify that the Project report, "**V-Care Disease Prediction System"** is the bonafide work of **ASHISH WILSON (Reg.No:AJC20MCA-2028)** in partial fulfillment of the requirements for the award of the Degree of Integrated Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2021-22.

Mr. Ajith G.S
Internal Guide

Ms. Nimmy Francis
Coordinator

Rev.Fr.Dr.Rubin Thottupurathu Jose

Head of the Department

DECLARATION

I hereby declare that the project report **"V-CARE DISEASE PREDICTION SYSTEM"** is a bonafided work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2021-2022.

Date: 22/02/2022 **ASHISH WILSON**

KANJIRAPPALLY **Reg. No: AJC20MCA-2028**

ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department fdgfgdgfdg **Rev. Fr. Dr. Rubin Thottupurathu Jose** for helping us. I extend my whole hearted thanks to the project coordinator **Ms**. **Nimmy Francis** for their valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also like to express sincere gratitude to my guide**, Mr**. **Ajith G.S** for his inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

ASHISH WILSON

ABSTRACT

The recent days have been hard for everyone due to the COVID-19 pandemic and a lot of people are scared to even step out of their homes due to this. Hence, people are less likely to visit a doctor these days since they have to go out of their homes and visit the doctor. The doctor then might suggest another specialist depending on the severity and the type of the disease.

Hence, a more accessible method is required for people to get checked if they have a specific disease. In India, especially, there has been a surge of internet usage in recent times and a lot more people now have access to smartphones and other devices through which they can access the internet easily.
Therefore, we will be creating a system that helps the people know what kind of disease they might have instead of directly using search engines which could generally give out wrong data due to inappropriate searches of symptoms by people.

This application will assess the information given by the user by asking specific questions and their severity and then predict what disease they might have.

CONTENT

Sl. No	Topic	Page No
1	INTRODUCTION	1
1.1	PROJECT OVERVIEW	2
1.2	PROJECT SPECIFICATION	2
2	SYSTEM STUDY	3
2.1	INTRODUCTION	4
2.2	EXISTING SYSTEM	5
2.3	DRAWBACKS OF EXISTING SYSTEM	5
2.4	PROPOSED SYSTEM	5
2.5	ADVANTAGES OF PROPOSED SYSTEM	6
3	REQUIREMENT ANALYSIS	7
3.1	FEASIBILITY STUDY	8
3.1.1	ECONOMICAL FEASIBILITY	8
3.1.2	TECHNICAL FEASIBILITY	9
3.1.3	BEHAVIORAL FEASIBILITY	9
3.2	SYSTEM SPECIFICATION	10
3.2.1	HARDWARE SPECIFICATION	10
3.2.2	SOFTWARE SPECIFICATION	10
3.3	SOFTWARE DESCRIPTION	10
3.3.1	PHP	10
3.3.2	MYSQL	11
4	SYSTEM DESIGN	13
4.1	INTRODUCTION	14
4.2	UML DIAGRAM	14
4.2.1	USE CASE DIAGRAM	15
4.2.2	SEQUENCE DIAGRAM	18
4.5	USER INTERFACE DESIGN	21
4.6	DATA BASE DESIGN	26
5	SYSTEM TESTING	32
5.1	INTRODUCTION	33
5.2	TEST PLAN	34

5.2.1	**UNIT TESTING**	**34**
5.2.2	**INTEGRATION TESTING**	**35**
5.2.3	**VALIDATION TESTING**	**35**
5.2.4	**USER ACCEPTANCE TESTING**	**36**
6	**IMPLEMENTATION**	**37**
6.1	**INTRODUCTION**	**38**
6.2	**IMPLEMENTATION PROCEDURE**	**38**
6.2.1	**USER TRAINING**	**39**
6.2.2	**TRAINING ON APPLICATION SOFTWARE**	**39**
6.2.3	**SYSTEM MAINTENANCE**	**39**
7	**CONCLUSION & FUTURE SCOPE**	**40**
7.1	**CONCLUSION**	**41**
7.2	**FUTURE SCOPE**	**41**
8	**BIBLIOGRAPHY**	**42**
9	**APPENDIX**	**44**
9.1	**SAMPLE CODE**	**45**
9.2	**SCREEN SHOTS**	**68**

List of Abbreviation

IDE	-	Integrated Development Environment
HTML	-	Hyper Text Markup Language.
CSS	-	Cascading Style Sheet
SQL	-	Structured Query Language
UML	-	Unified Modeling Language

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

Most of the people tend to search on search engines like Google to find out what disease they might have. This could be effective, but sometimes, the search engine might show a different conclusion since the severity of some symptoms is not mentioned by the user. Therefore, this outcome could be inaccurate, it could show something very severe or minor, both of which could be harmful for the user and might cause him/her to panic.

The project focuses on prediction of disease basis on the symptoms entered by the users also helping them by giving precautions.The prediction is based on the input entered by the users.

1.2 PROJECT SPECIFICATION

The proposed system is a website in which user can predict there disease based on symptoms. Also doctors are able to add new disease and symptoms into the knowledge base .

The system includes 3 modules. They are:

1. Admin Module

Admin must have a login into this system. He has the overall control of the system. Admin, manage user data etc. Admin can View all the registered users and also manage all his data.

2.User

User will register their name and their details in this app via registration page.They can sign in the login page on successful registration. The user has right to edit and view his/her profile details.User can add their symptoms and based on the symptoms in the database, the disease will be predicted.

3.Doctor

Doctor will register their name and their details in this app via registration page. They can sign in the login page on successful registration. The doctor can add new disease and their symptoms and it can viewed by the users

CHAPTER 2

SYSTEM STUDY

2.1 INTRODUCTION

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements on the system. It is a problem solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minute's detail and analyzed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system are identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analyzing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action.

A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal.

Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies, a rough figure of the system activities can be obtained, from which the decision about the strategies to be followed for effective system study and analysis can be taken.

2.2 EXISTING SYSTEM

Most of the people tend to search on search engines like Google to find out what disease they might have. This could be effective, but sometimes, the search engine might show a different conclusion since the severity of some symptoms is not mentioned by the user. Therefore, this outcome could be inaccurate, it could show something very severe or minor, both of which could be harmful for the user and might cause him/her to panic. There is no proper disease prediction system for users

2.3 DRAWBACKS OF EXISTING SYSTEM

- No proper disease prediction system
- Human effort is needed.
- It is difficult for old people to vist hospitals for minor diseases .

2.4 PROPOSED SYSTEM

The proposed V Care Online Disease Prediction system would help a lot more people self-diagnose so that they can take appropriate steps to counter the situations. For example, once the application gives an output mentioning a disease, the user might directly contact a specialist if needed instead of going to a general physician. In later versions, we could also use this data to make this system better and also help the people get directions to the nearest doctor/specialist.

The system helps users by giving a percentage of predictions also it gives a clear picture of his/her conditions.The prediction system also help old people also as it is not necessary to vist a clinic for minor diseases.

The prediction of disease is based on the symptoms entered by the users and it will giving appropriate prediction based on values entered by doctors.Doctors have an option to edit symptom details also values.The prediction is completely depends on the symptoms entered by the users.

2.5 ADVANTAGES OF PROPOSED SYSTEM

The system is very simple in design and to implement. The system requires very low system resources, and the system will work in almost all configurations. It has got following features:

- **Better security: -**

For data to remain secure measures must be taken to prevent unauthorized access. Security means that data are protected from various forms of destruction. The system security problem can be divided into four related issues: security, integrity, privacy and confidentiality. Username and password requirement to sign in ensures security. It will also provide data security as we are using the secured databases for maintaining the documents.

- **Ensure data accuracy: -**

The proposed system eliminates the manual errors while entering the details of the users during the registration.

- **Better service: -**

The product will avoid the burden of hard copy storage. We can also conserve the time and human resources for doing the same task. The data can be maintained for longer period with no loss of data.

- **Prediction of disease: -**

Using V Care Disease prediction system diseases will be predicted with the help of symptoms entered by the user and give a prediction on basis of symptoms also the system will help users with precautions.

CHAPTER 3

REQUIREMENT ANALYSIS

3.1 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 spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its workability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Thus, when a new application 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, Economic and Operational feasibilities. The following are its features: -

3.1.1 Economical 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.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

The proposed system is 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.

The cost of project, DREAMS was divided according to the system used, its development cost and cost for hosting the project. According to all the calculations the project was developed in a low cost. As it is completely developed using open source software.

3.1.2 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 and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed.

Technical issues raised during the investigation are:

- Does the existing technology sufficient for the suggested one?
- Can the system expand if developed?

The project should be developed such that the necessary functions and performance are achieved within the constraints. The project requires High Resolution Scanning device and utilizes Cryptographic techniques. Through the technology may become obsolete after some period of time, due to the fact that newer version of same software supports older versions, the system may still be used. So there are minimal constraints involved with this project. The system has been developed using PHP in front end and MySQL in server in back end, the project is technically feasible for development. The system has been developed using PHP in front end and MySQL in server in back end, the project is technically feasible for development. The System used was also of good performance of Processor Intel i3 core; RAM 4GB and, Hard disk 1TB

3.1.3 Behavioral Feasibility

The proposed system includes the following questions:

- Is there sufficient support for the users?
- Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

3.2 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor - Intel core i3

RAM - 4 GB

Hard disk - 1 TB

3.2.2 Software Specification

Front End - HTML, CSS

Backend - MYSQL

Client on PC - Windows 7 and above.

Technologies used - JS, HTML5, AJAX, J Query, PHP, CSS

3.3 SOFTWARE DESCRIPTION

3.3.1 PHP

PHP is a server side scripting language designed for web development but also used as a general purpose programming language. PHP is now installed on more than 244 million websites and 2.1 million web servers. Originally created by Rasmus Ledorf in 1995, the reference implementation of PHP is now produced by the PHP group. While PHP originally stood for personal Home page ,it now stands for PHP:HypertextPreprocessor, a recursive acronym.PHP code is interpreted by a web server with a PHP processor module which generates the resulting web page.PHP commands can be embedded directly into a HTML source document rather than calling an external file to process data. It has also evolved to include a command-line interface capability and can be used in standalone incompatible with the GNU General Public License (GPL) due to restrictions on the usage of the term PHP.PHP can be deployed on most web servers and also as a standalone shell on almost every operating system and platform, free of charge.

3.3.2 MySQL

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. The MySQL Web site provides the latest information about MySQL software.

- **MySQL is a database management system.**

  A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- **MySQL databases are relational.**

  A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and "pointers" between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data. The SQL part of "MySQL" stands for "Structured Query Language". SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax. SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, "SQL92" refers to the standard released in 1992,

  "SQL: 1999" refers to the standard released in 1999, and "SQL: 2003" refers to the current version of the standard. We use the phrase "the SQL standard" to mean the current version of the SQL Standard at any time.

- **MySQL software is Open Source.**

  Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information.

- **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

  If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available.

- **MySQL Server works in client/server or embedded systems.**

  The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs). We also provide MySQL Server as an embedded multi-threaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

CHAPTER 4

SYSTEM DESIGN

4.1 INTRODUCTION

Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. The term "design" is defined as "the process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization". It may be defined as a process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization. Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm that is used. The system design develops the architectural detail required to build a system or product. As in the case of any systematic approach, this software too has undergone the best possible design phase fine tuning all efficiency, performance and accuracy levels. The design phase is a transition from a user oriented document to a document to the programmers or database personnel. System design goes through two phases of development: Logical and Physical Design.

4.2 UML DIAGRAM

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997.

UML stands for **Unified Modeling Language**. UML is different from the other common programming languages such as C++, Java, COBOL, etc. UML is a pictorial language used to make software blueprints. UML can be described as a general purpose visual modeling language to visualize, specify, construct, and document software system. Although UML is generally used to model software systems, it is not limited within this boundary. It is also used to model non-software systems as well. For example, the process flow in a manufacturing unit, etc. UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object oriented analysis and design. After some standardization, UML has

become an OMG standard. All the elements, relationships are used to make a complete UML diagram and the diagram represents a system. The visual effect of the UML diagram is the most important part of the entire process. All the other elements are used to make it complete. UML includes the following nine diagrams.

- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Collaboration diagram
- Activity diagram
- Statechart diagram
- Deployment diagram
- Component diagram

4.2.1 USE CASE DIAGRAM

A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. In this context, the term "system" refers to something being developed or operated, such as a mail-order product sales and service Web site. Use case diagrams are employed in UML (Unified Modeling Language), a standard notation for the modeling of real-world objects and systems.

System objectives can include planning overall requirements, validating a hardware design, testing and debugging a software product under development, creating an online help reference, or performing a consumer-service-oriented task. For example, use cases in a product sales environment would include item ordering, catalog updating, payment processing, and customer relations. A use case diagram contains four components.

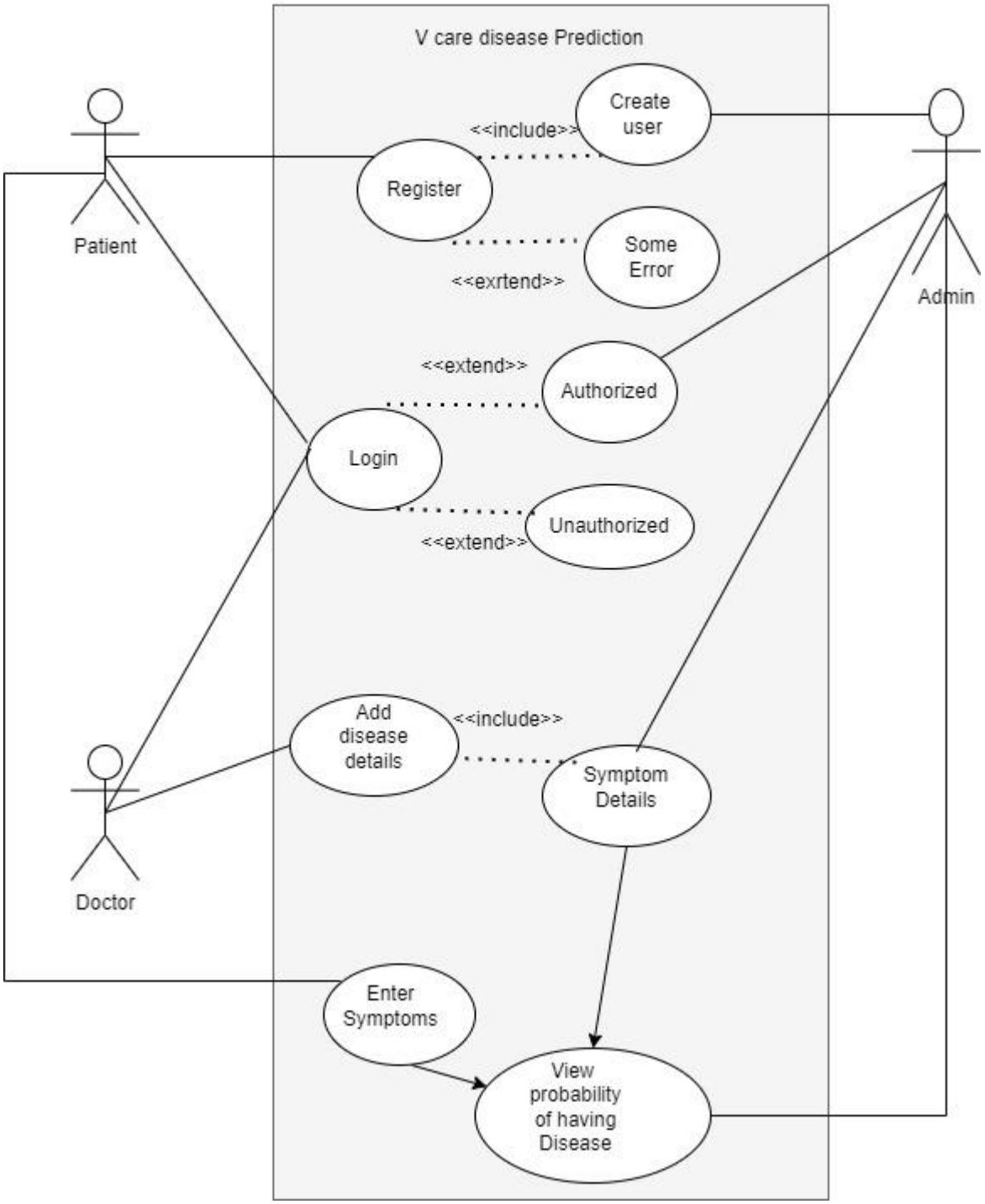
- The boundary, which defines the system of interest in relation to the world around it.
- The actors, usually individuals involved with the system defined according to their roles.
- The use cases, which are the specific roles are played by the actors within and around the system.

- The relationships between and among the actors and the use cases.

Use case diagrams are drawn to capture the functional requirements of a system. After identifying the above items, we have to use the following guidelines to draw an efficient use case diagram

- The name of a use case is very important. The name should be chosen in such a way so that it can identify the functionalities performed.
- Give a suitable name for actors.
- Show relationships and dependencies clearly in the diagram.
- Do not try to include all types of relationships, as the main purpose of the diagram is to identify the requirements.
- Use notes whenever required to clarify some important points.

Use Case Diagram



4.2.2 SEQUENCE DIAGRAM

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

Sequence Diagram Notations –

i. **Actors –** An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram. We use actors to depict various roles including human users and other external subjects. We represent an actor in a UML diagram using a stick person notation. We can have multiple actors in a sequence diagram.

ii. **Lifelines –** A lifeline is a named element which depicts an individual participant in a sequence diagram. So basically each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram

iii. **Messages –** Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows. Lifelines and messages form the core of a sequence diagram.

Messages can be broadly classified into the following categories:

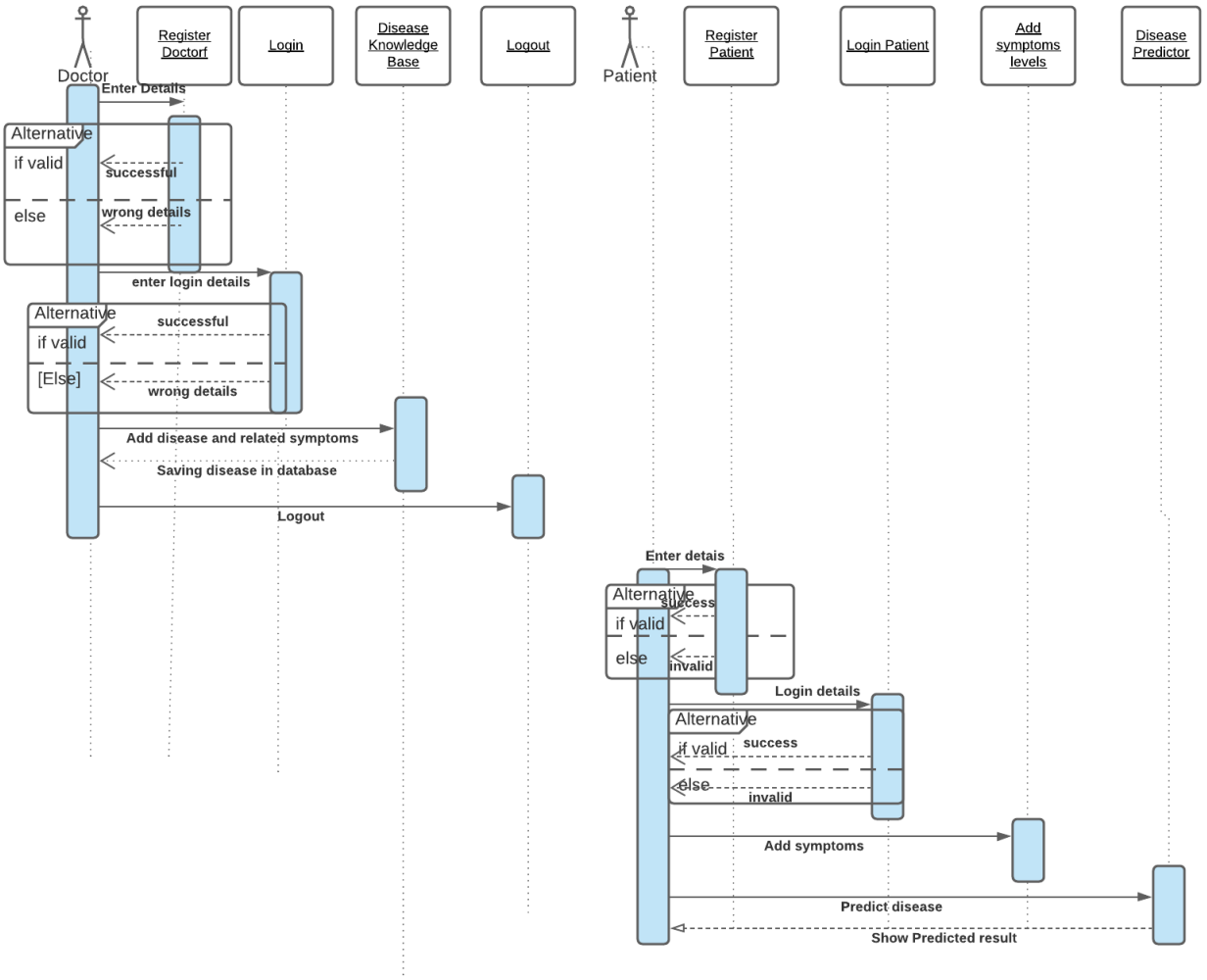
- Synchronous messages
- Asynchronous Messages
- Create message
- Delete Message
- Self-Message
- Reply Message

- Found Message
- Lost Message

iv. **Guards –** To model conditions we use guards in UML. They are used when we need to restrict the flow of messages on the pretext of a condition being met. Guards play an important role in letting software developers know the constraints attached to a system or a particular process.

Uses of sequence diagrams –

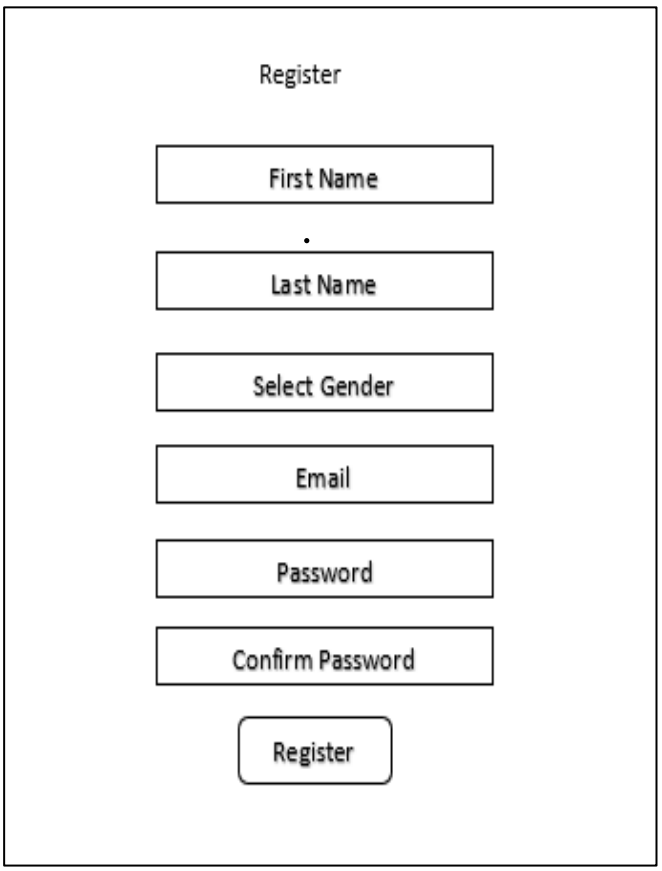
- Used to model and visualize the logic behind a sophisticated function, operation or procedure.
- They are also used to show details of UML use case diagrams.
- Used to understand the detailed functionality of current or future systems.
- Visualize how messages and tasks move between objects or components in a system.


Doctor
Register Doctorf
Login
Disease Knowledge Base
Logout
Patient
Register Patient
Login Patient
Add symptoms levels
Disease Predictor
Enter Details
Alternative
if valid
successful
else
wrong details
enter login details
Alternative
if valid
successful
[Else]
wrong details
Add disease and related symptoms
Saving disease in database
Logout
Enter detais
Alternative
if valid
success
else
invalid
Login details
Alternative
if valid
success
else
invalid
Add symptoms
Predict disease
Show Predicted result

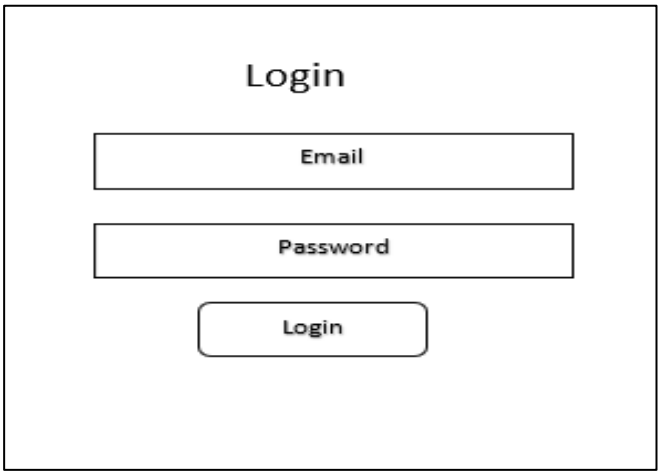
4.5 USER INTERFACE DESIGN

4.5.1-INPUT DESIGN

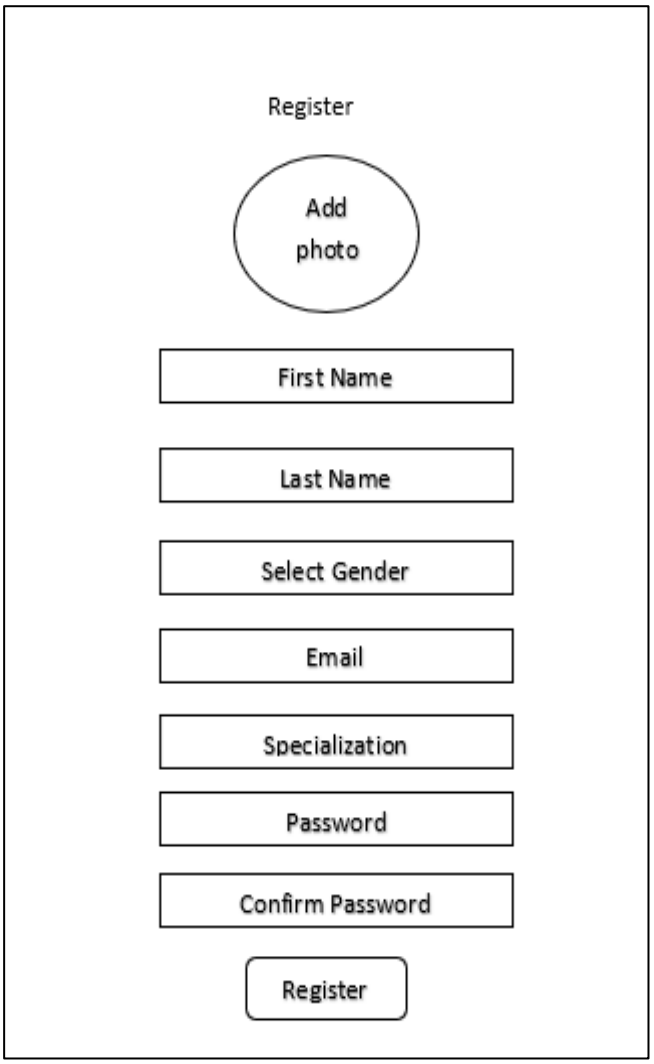
Form Name : User Registration



Form Name : Login

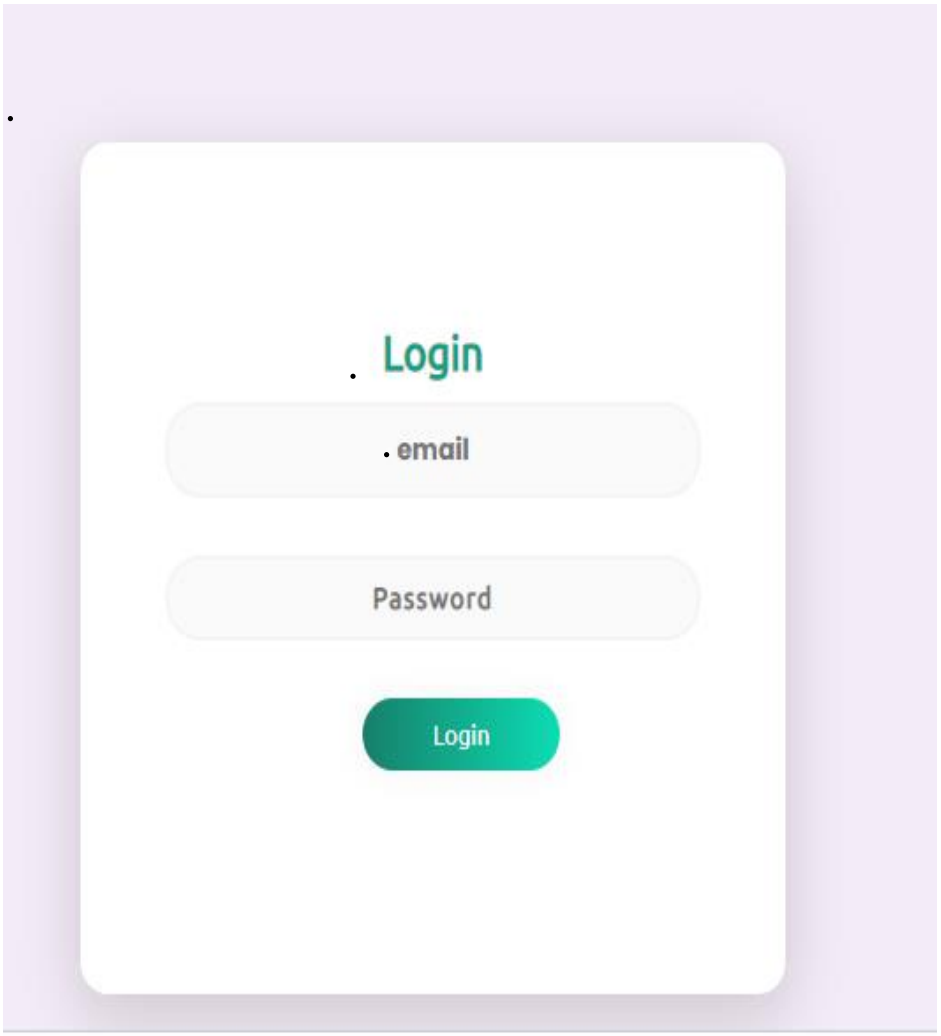


Form Name : Doctor Register Page

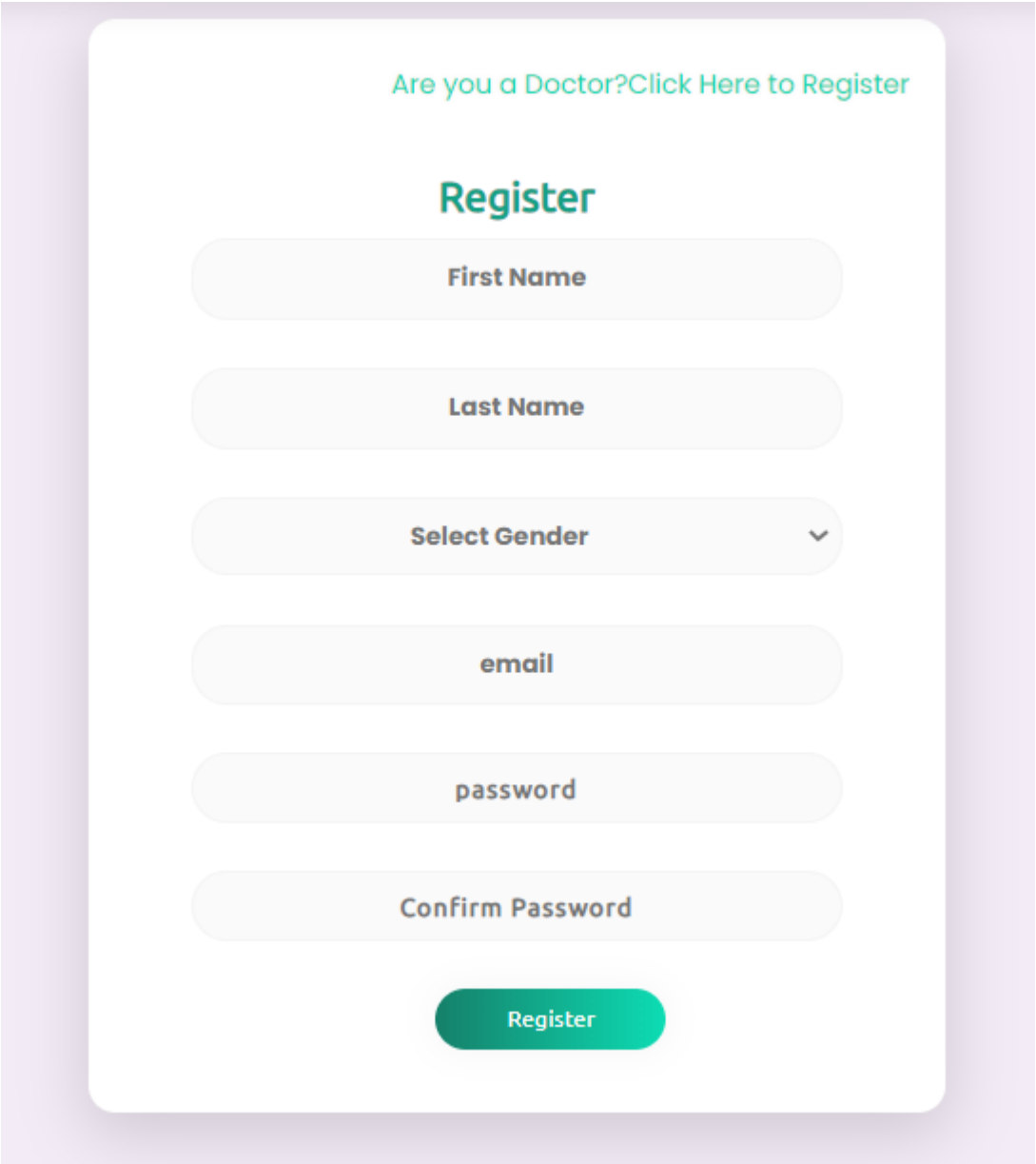


4.5.2 OUTPUT DESIGN

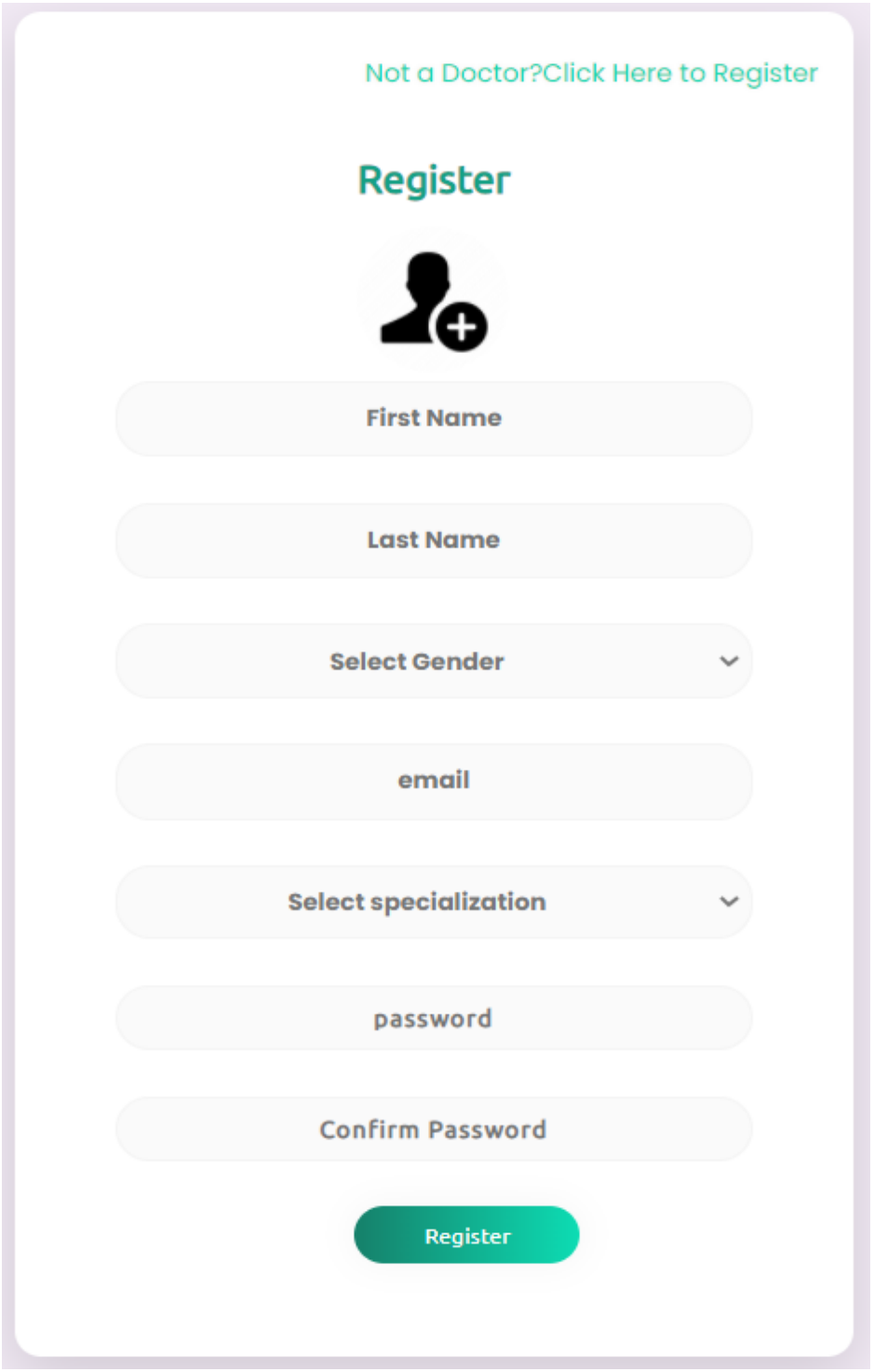
Login



User Registration



Doctor Registration



4.6. DATABASE DESIGN

A database is an organized mechanism that has the capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected.

The database design is a two level process. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS.

In the second step, this Information level design is transferred into a design for the specific DBMS that will be used to implement the system in question. This step is called Physical Level Design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following two major objectives.

- Data Integrity
- Data independence

4.6.1 Relational Database Management System (RDBMS)

A relational model represents the database as a collection of relations. Each relation resembles a table of values or file of records. In formal relational model terminology, a row is called a tuple, a column header is called an attribute and the table is called a relation. A relational database consists of a collection of tables, each of which is assigned a unique name. A row in a tale represents a set of related values.

Relations, Domains & Attributes

A table is a relation. The rows in a table are called tuples. A tuple is an ordered set of n elements. Columns are referred to as attributes. Relationships have been set between every table in the database. This ensures both Referential and Entity Relationship Integrity. A domain D is a set of atomic values. A common method of specifying a domain is to specify a data type from which the data values forming the domain are drawn. It is also useful to specify a name for the domain to help in interpreting its values.

Every value in a relation is atomic, that is not decomposable.

Relationships

- Table relationships are established using Key. The two main keys of prime importance are Primary Key & Foreign Key. Entity Integrity and Referential Integrity Relationships can be established with these keys.
- Entity Integrity enforces that no Primary Key can have null values.
- Referential Integrity enforces that no Primary Key can have null values.
- Referential Integrity for each distinct Foreign Key value, there must exist a matching Primary Key value in the same domain. Other key are Super Key and Candidate Keys.

4.6.2 Normalization

Data are grouped together in the simplest way so that later changes can be made with minimum impact on data structures. Normalization is formal process of data structures in manners that eliminates redundancy and promotes integrity. Normalization is a technique of separating redundant fields and breaking up a large table into a smaller one. It is also used to avoid insertion, deletion, and updating anomalies. Normal form in data modelling use two concepts, keys and relationships. A key uniquely identifies a row in a table. There are two types of keys, primary key and foreign key. A primary key is an element or a combination of elements in a table whose purpose is to identify records from the same table. A foreign key is a column in a table that uniquely identifies record from a different table. All the tables have been normalized up to the third normal form.

As the name implies, it denotes putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and thereby avoids data redundancy which proves to be a great burden on the computer resources. These include:

- ✓ Normalize the data.
- ✓ Choose proper names for the tables and columns.
- ✓ Choose the proper name for the data.

First Normal Form

The First Normal Form states that the domain of an attribute must include only atomic values and that the value of any attribute in a tuple must be a single value from the domain of that attribute. In other words 1NF disallows “relations within relations” or “relations as attribute values within tuples”. The only attribute values permitted by 1NF are single atomic or indivisible values. The first step is to put the data into First Normal Form. This can be donor by moving data into separate tables where the data is of similar type in each table. Each table is given a Primary Key or Foreign Key as per requirement of the project. In this we form new relations for each non-atomic attribute or nested relation. This eliminated repeating groups of data. A relation is said to be in first normal form if only if it satisfies the constraints that contain the primary key only.

Second Normal Form

According to Second Normal Form, for relations where primary key contains multiple attributes, no non-key attribute should be functionally dependent on a part of the primary key. In this we decompose and setup a new relation for each partial key with its dependent attributes. Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it. This step helps in taking out data that is only dependent on a part of the key. A relation is said to be in second normal form if and only if it satisfies all the first normal form conditions for the primary key and every non-primary key attributes of the relation is fully dependent on its primary key alone.

Third Normal Form

According to Third Normal Form, Relation should not have a non-key attribute functionally determined by another non-key attribute or by a set of non-key attributes. That is, there should be no transitive dependency on the primary key. In this we decompose and set up relation that includes the non-key attributes that functionally determines other non-key attributes. This step is taken to get rid of anything that does not depend entirely on the Primary Key. A relation is said to be in third normal form if only if it is in second normal form and more over the non key attributes of the relation should not be depend on other non-key attribute.

TABLE DESIGN

Table 1 : Login

Primary key : Login_id

Table Description : To store user login details

Column name	Datatype	Constraints	Description
Login_id	Int	Primary key	Primary key of the table
Email	Varchar(35)	Not Null	Unique key
Password	Varchar(25)	Not Null	Password of the user
Role	Varchar(10)	Not null	Role of the user
Status	Int	Not null	Status of the user

Table 2 : Patient Registration

Primary key : user_id

Foreign Key : login_id

Table Description : To store user patient details

Column name	Datatype	Constraints	Description
user_id	Int	Primary key	Primary key of the table
First name	Varchar(25)	Not Null	Firstname of patient
Last name	Varchar(25)	Not Null	Lastname of patient
Gender	Varchar(10)	Not Null	Gender of patient
Phone Number	Int	Not Null	Phoneneumber of patient
Login_id	Int	Foreign key	Loginid of patient

Table 3 : Doctor _Registration

Primary key : doc_id

Foreign Key : login_id

Table Description : To store user doctor details

Column name	Datatype	Constraints	Description
doc_id	int	Primary key	Primary Key of Doctor table
First name	Varchar(25)	Not Null	Firstname of doctor
Last name	Varchar(25)	Not Null	Lastname of doctor
Gender	Varchar(10)	Not Null	Gender of Doctor
Doctor_licence_no	Varchar(25)	Not Null	Doctor's License number
Phone number	int	Not Null	Phone number of doctor
Login_id	Int	Foreign key	LoginId of Doctor

Table 4 : Symptom

Primary key : symptom_id

Foreign Key : login_id

Table Description : To store symptom details

Column name	Datatype	Constraints	Description
Symptom_id	int	Primary key	Primary Key of symptom table
Symptom _name	Varchar(30)	Not Null	Name of symptom
Fuzzy set	Varchar(50)	Not Null	Fuzzy set values of symptom
Range_value	int	Not Null	Rangeof symptoms
Description	Varchar(500)	Not Null	Description of symptom ranges
Login_id	Int	Foreign key	Login Id of Doctor

Table 5 :Mapping

Primary key : disease_id, symptom_id

Foreign Key :

Table Description : To store mapping details

Column name	Datatype	Constraints	Description
disease_id	int	Primary key	Composite Key of the table
Fuzzyvalue	Varchar(20)	Not Null	Fuzzy values of the symptom
Weight	Varchar(30)	Not Null	Weightage of symptom
symptom_id	int	Foreign key	Composite Key of symptom

Table 6 : Disease

Primary key : Disease_name

Foreign key : login_id,disease_id

Table Description : To store disease details

Column name	Datatype	Constraints	Description
Disease_name	Varchar(25)	Primary key	Primary Key of the table
Specialist	Varchar(20)	Not Null	Specailsit of the disease
Precaution	Varchar(30)	Not Null	Precaution of the disease
Disease_id	int	Foreign Key	Id of the disease
Login_id	Int	Foreign key	LoginId of the doctor

Table 7 : Feedback

Primary key : Feedback_id

Foreign key : login_id

Table Description : To store feedback details

Column name	Datatype	Constraints	Description
Feedback_id	int	Primary Key	Primary Key of the table
Rating	int	Not Null	Rating of the doctor
Review	Varchar(20)	Not Null	Review of the doctor service
login_id	int	Foreign key	Login id of the patient

Table 8 : Diag

Primary key : index(x)

Foreign key :

Table Description : To store fuzzy set algorithm details for calculating the diseases

Column name	Datatype	Constraints	Description
x	int	Primary Key	Primary Key-index of the table
yes	int	Not Null	Value of Yes possibilty
Maybe	int	Not Null	Value of Maybe possibilty
No	int	Not Null	Value of No possibilty

CHAPTER 5

SYSTEM TESTING

5.1 INTRODUCTION

Software Testing is the process of executing software in a controlled manner, in order to answer the question - Does the software behave as specified? Software testing is often used in association with the terms verification and validation. Validation is the checking or testing of items, includes software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, analysis, inspections, and walkthroughs. Validation is the process of checking that what has been specified is what the user actually wanted.

Other activities which are often associated with software testing are static analysis and dynamic analysis. Static analysis investigates the source code of software, looking for problems and gathering metrics without actually executing the code. Dynamic analysis looks at the behavior of software while it is executing, to provide information such as execution traces, timing profiles, and test coverage information.

Testing is a set of activity that can be planned in advanced and conducted systematically. Testing begins at the module level and work towards the integration of entire computers based system. Nothing is complete without testing, as it vital success of the system testing objectives, there are several rules that can serve as testing objectives. They are:

Testing is a process of executing a program with the intent of finding an error.

- A good test case is one that has high possibility of finding an undiscovered error.
- A successful test is one that uncovers an undiscovered error.

If a testing is conducted successfully according to the objectives as stated above, it would uncover errors in the software. Also testing demonstrate that the software function appear to be working according to the specification, that performance requirement appear to have been met.

There are three ways to test program.

- For correctness
- For implementation efficiency
- For computational complexity

Test for correctness are supposed to verify that a program does exactly what it was designed to do. This is much more difficult than it may at first appear, especially for large programs.

5.2 TEST PLAN

A test plan implies a series of desired course of action to be followed in accomplishing various testing methods. The Test Plan acts as a blue print for the action that is to be followed. The software engineers create a computer program, its documentation and related data structures. The software developers is always responsible for testing the individual units of the programs, ensuring that each performs the function for which it was designed. There is an independent test group (ITG) which is to remove the inherent problems associated with letting the builder to test the thing that has been built. The specific objectives of testing should be stated in measurable terms. So that the mean time to failure, the cost to find and fix the defects, remaining defect density or frequency of occurrence and test work-hours per regression test all should be stated within the test plan.

The levels of testing include:

- Unit testing
- Integration Testing
- Data validation Testing
- Output Testing

5.2.1 Unit Testing

Unit testing focuses verification effort on the smallest unit of software design – the software component or module. Using the component level design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The relative complexity of tests and uncovered scope established for unit testing. The unit testing is white-box oriented, and step can be conducted in parallel for multiple components. The modular interface is tested to ensure that information properly flows into and out of the program unit under test. The local data structure is examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithm's execution. Boundary conditions are tested to ensure that all statements in a module have been executed at least once. Finally, all error handling paths are tested.

Tests of data flow across a module interface are required before any other test is initiated. If data do not enter and exit properly, all other tests are moot. Selective testing of execution paths is an essential task during the unit test. Good design dictates that error conditions be anticipated and error handling paths set up to reroute or cleanly terminate processing when an error does occur. Boundary testing is the last task of unit testing step. Software often fails at its boundaries.

Unit testing was done in Sell-Soft System by treating each module as separate entity and testing each one of them with a wide spectrum of test inputs. Some flaws in the internal logic of the modules were found and were rectified. After coding each module is tested and run individually. All unnecessary code where removed and ensured that all modules are working, and gives the expected result.

5.2.2 Integration Testing

Integration testing is systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective is to take unit tested components and build a program structure that has been dictated by design. The entire program is tested as whole. Correction is difficult because isolation of causes is complicated by vast expanse of entire program. Once these errors are corrected, new ones appear and the process continues in a seemingly endless loop. After performing unit testing in the System all the modules were integrated to test for any inconsistencies in the interfaces. Moreover differences in program structures were removed and a unique program structure was evolved.

5.2.3 Validation Testing or System Testing

This is the final step in testing. In this the entire system was tested as a whole with all forms, code, modules and class modules. This form of testing is popularly known as Black Box testing or System tests.

Black Box testing method focuses on the functional requirements of the software. That is, Black Box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program.

Black Box testing attempts to find errors in the following categories; incorrect or missing functions, interface errors, errors in data structures or external data access, performance errors and initialization errors and termination errors.

5.2.4 Output Testing or User Acceptance Testing

The system considered is tested for user acceptance; here it should satisfy the firm's need. The software should keep in touch with perspective system; user at the time of developing and making changes whenever required. This done with respect to the following points:

- Input Screen Designs,
- Output Screen Designs,

The above testing is done taking various kinds of test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

CHAPTER 6

IMPLEMENTATION

6.1 INTRODUCTION

Implementation is the stage of the project where the theoretical design is turned into a working system. It can be considered to be the most crucial stage in achieving a successful new system gaining the users confidence that the new system will work and will be effective and accurate. It is primarily concerned with user training and documentation. Conversion usually takes place about the same time the user is being trained or later. Implementation simply means convening a new system design into operation, which is the process of converting a new revised system design into an operational one.

At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned or controlled, it can create chaos and confusion.

Implementation includes all those activities that take place to convert from the existing system to the new system. The new system may be a totally new, replacing an existing manual or automated system or it may be a modification to an existing system. Proper implementation is essential to provide a reliable system to meet organization requirements. The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from the old system to the new system. The system can be implemented only after through testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system. The more complex the system being implemented, the more involved will be the system analysis and design effort required to implement the three main aspects: education and training, system testing and changeover.

The implementation state involves the following tasks:

- Careful planning.
- Investigation of system and constraints.
- Design of methods to achieve the changeover.

6.2 IMPLEMENTATION PROCEDURES

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended uses and the operation of the system. In many organizations someone who will not be operating it, will commission the software development project. In the initial stage people doubt about the software but we have to

ensure that the resistance does not build up, as one has to make sure that:

- The active user must be aware of the benefits of using the new system.
- Their confidence in the software is built up.
- Proper guidance is imparted to the user so that he is comfortable in using the application.

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not up running on the server, the actual process won't take place.

6.2.1 User Training

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer based system, it is essential for the people who will be involved to be confident of theirrole in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database and call up routine that will produce reports and perform other necessary functions.

6.2.2 Training on the Application Software

After providing the necessary basic training on computer awareness the user will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the date entered. It should then cover information needed by the specific user/ group to use the system or part of the system while imparting the training of the program on the application. This training may be different across different user groups and across different levels of hierarchy

6.2.3 System Maintenance

Maintenance is the enigma of system development. The maintenance phase of the software cycle is the time in which a software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is for it to make adaptable to the changes in the system environment. Software maintenance is of course, far more than "Finding Mistakes".

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The current system working technology is old fashioned and there is no usage of commonly used technologies like internet, digital money. The proposed system introduces facility for customer to book service online and view all information.

7.2 FUTURE SCOPE

- The proposed system is designed in such a way that the patients can do video appointment with the docto and there by the rush in hosptials can be reduced
- Patient can shop medicines in online system.
- Instant appointment will be added to this system..

.

CHAPTER 8

BIBLIOGRAPHY

REFERENCES:

- Gary B. Shelly, Harry J. Rosenblatt, "*System Analysis and Design*", 2009.
- Roger S Pressman, "*Software Engineering*", 1994.
- PankajJalote, "So*ftware engineering*: a precise approach", 2006.
- James lee and Brent ware Addison, "Open source web development with LAMP", 2003
- IEEE Std 1016 Recommended Practice for Software Design Descriptions.

WEBSITES:

- www.w3schools.com
- www.jquery.com
- http://homepages.dcc.ufmg.br/~rodolfo/es-1-03/IEEE-Std-830-1998.pdf
- www.agilemodeling.com/artifacts/useCaseDiagram.html

CHAPTER 9

APPENDIX

9.1 Sample Code

Login.php

```
<html>
<head>
<link rel="stylesheet" href="css/login.css">
<link rel="stylesheet" href="css/style.css">

<link href="https://fonts.googleapis.com/css?family=Ubuntu" rel="stylesheet">
<meta name="viewport" content="width=device-width, initial-scale=1" />
<link rel="stylesheet" href="path/to/font-awesome/css/font-awesome.min.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.9.0/jquery.min.js"></script>
<title>Sign in</title>
</head>
<?php
session_start();

include 'header.php';
include 'config.php';
?>

<body>
<section class="tab">
<div class="clearfix secondary-nav" style="margin-left:627px;margin-top:65px;">
<ul class="list">
<li><a style="text-decoration : underline;color: #16a085;" class="logintab"
href="login.php">Login</a></li>
<li><a class="rtab" href="uregister.php">Register</a></li>
</ul>
</div>
</section>

<section class="home loginpage" id="home">
<div class="loginimage" align="center">
<img src="image/online-doctor-animate (1).svg" alt="">
</div>
<div class="main" align="center">
<p class="sign" align="center">Login</p>
<form name="form1" class="form1" method="POST" onsubmit="return validate()">
<div class="id" id="mismatchError"></div>
<input class="un" type="email" id="email" name="email" align="center" placeholder="email">
<div class="id" id="nameErr"></div>
<input class="pass" type="password" id="pass" name="pass" align="center"
placeholder="Password">
<div class="id" id="passErr"></div>
<input class="submit" type="submit" name="submit" align="center" value="Login">
<!-- <p class="forgot" align="center"><a style="margin-left: 38px; " href="#">Forgot Password?</p>
-->
</form>
</div>
</section>
```

```
<script>
function validate() {
if (document.form1.email.value.trim() == "") {
printError("nameErr", "Please enter email address");
document.getElementById("email").classList.add("errorClass");
// return false;
}
else {
printNoError("nameErr", "");
document.getElementById("email").classList.remove("errorClass");
}


if (document.form1.pass.value.trim() == "") {
printError("passErr", "Please enter password");
document.getElementById("pass").classList.add("errorClass");
//   return false;

}
else {
printNoError("passErr", "");
document.getElementById("pass").classList.remove("errorClass");
}

if (document.form1.email.value.trim() == "" || document.form1.pass.value.trim() == "") {
return false;
} else {
return true;
}
}

function printError(elemId, hintMsg) {
var elem = document.getElementById(elemId);
elem.classList.add("error")
elem.innerHTML = hintMsg;
}
function printNoError(elemId, hintMsg) {
var elem = document.getElementById(elemId);
elem.classList.remove("error");
elem.innerHTML = hintMsg;
}
function errorMsg(hintMsg){
var elem = document.getElementById("mismatchError");
elem.classList.add("error")
elem.innerHTML = hintMsg;
document.getElementById("email").classList.add("errorSub");
document.getElementById("pass").classList.add("errorSub");

}

</script>
<?php
if (isset($_POST['submit'])) {
$user = $_POST['email'];
```

```
$pass = $_POST['pass'];

$sql = mysqli_query($con, "SELECT * FROM login WHERE email='$user' AND pass='$pass'");
if (mysqli_num_rows($sql) > 0) {
while ($row = mysqli_fetch_assoc($sql)) {
if ($row['sta'] == 1) {
if ($row['role'] == 'admin') {
$_SESSION['UserID'] = $row['loginid'];
header("location: admin");
} else if ($row['role'] == 'user') {
$_SESSION['UserID'] = $row['loginid'];
header("location: patient");
} else if ($row['role'] == 'doc') {
$_SESSION['UserID'] = $row['loginid'];
header("location: doctor");
}
} else {
?>
<script>errorMsg('User is blocked by admin')
</script>
<?php  }
}
} else { ?>
<script>
errorMsg('Username and password are incorrect');
</script>
<?php   }
}
?>
<?php include 'footer.php'; ?>
</body>
</html>
```

Config.php

```
<?php
$con = mysqli_connect("localhost","root","","miniprj")or die ("not connected ");
?>
```

Doctorregister.php

```
<html>

<head>
<link rel="stylesheet" href="css/login.css">
<link rel="stylesheet" href="css/style.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.9.0/jquery.min.js"></script>
<link href="https://fonts.googleapis.com/css?family=Ubuntu" rel="stylesheet">
<meta name="viewport" content="width=device-width, initial-scale=1" />
```

```
<title>Sign in</title>
</head>
<?php include'header.php'; ?>

<body>
<?php
include 'config.php';
?>
<section class="tab" >

<div class="clearfix secondary-nav" style="margin-left:627px;margin-top:65px;">
<ul class="list">
<li><a class="logintab" href="login.php">Login</a></li>
<li><a style="text-decoration : underline;color: #16a085;" class="rtab"
href="uregister.php">Register</a></li>
</ul>
</div>
</section>

<section class="home loginpage" id="home">

<div class="loginimage"   align ="center">
<img src="image/online-doctor-animate.svg" alt="">
</div>

<div class="main" id="main1"  align ="center">
<a class="doctorregister" href="uregister.php">Not a Doctor?Click Here to Register</a>
<p class="sign" align="center">Register</p>

<form name="form1" class="form1" method="POST" onsubmit="return validate()"
enctype="multipart/form-data">
<img src="image/profile.png" onClick="triggerClick()" id="pd"/>
<input type="file" name="pi" onChange="displayImage(this)"  id="pi" style="display:none;">
<div class="id" id="pf"></div>
<input name="fn" class="un " type="text" id="fn" align="center" placeholder="First Name">
<div class="id" id="fname"></div>
<input name="ln" class="un " type="text" id="ln" align="center" placeholder="Last Name">
<div class="id" id="lname"></div>
<!-- <input type="date" data-date-inline-picker="true" class="un" align="center" placeholder="Date
Of Birth"> -->
<!-- <label for="gender">Gender :</label> -->
<select name="gender" name="gender" id="gender" class="un" align="center" value="gender">
<option class="un " value="">Select Gender</option>
<option value="Male">Male</option>
<option value="Female">Female</option>
</select>
<div class="id" id="genderErr"></div>
<input name="email" class="un" type="email" id="email" align="center" placeholder="email">
<div class="id" id="emailError"></div>
<select name="spec" id="specialization"  class="un" align="center" value="specialization">
<option  class="un " name="spec" value="">Select specialization</option>
<option value="gp">General Physician</option>
<option value="cardio">Cardiologists</option>
<option value="ortho">Orthopedist</option>
</select>
```

```
<div class="id" id="specErr"></div>

<input name="pass" class="pass" id="pass1" type="text" align="center" placeholder="password">
<div class="id" id="passError1"></div>
<input name="conpass" class="pass" id="pass2" type="password" align="center"
placeholder="Confirm Password">
<div class="id" id="passError2"></div>
<input type="submit" name="submit" class="submit" value="Register">
<!-- <p class="forgot" align="center"><a href="#">Forgot Password?</p> -->


</div>

</section>
<script>
function validate() {
if (document.form1.pi.value.trim() == "") {
printError("pf", "Please insert profile pic");
document.getElementById("pi").classList.add("errorClass");
}
else {
printNoError("pf", "");
document.getElementById("fn").classList.remove("errorClass");
}

if (document.form1.fn.value.trim() == "") {
printError("fname", "Please enter first name");
document.getElementById("fn").classList.add("errorClass");
}
else {
printNoError("fname", "");
document.getElementById("fn").classList.remove("errorClass");
}

if (document.form1.ln.value.trim() == "") {
printError("lname", "Please enter Last name");
document.getElementById("ln").classList.add("errorClass");
}
else {
printNoError("lname", "");
document.getElementById("ln").classList.remove("errorClass");
}

if (document.form1.email.value.trim() == "") {
printError("emailError", "Please enter email");
document.getElementById("email").classList.add("errorClass");
}
else {
printNoError("emailError", "");
document.getElementById("email").classList.remove("errorClass");
}

if (document.getElementById("pass1").value.trim() == "") {
printError("passError1", "Please enter password");
document.getElementById("pass1").classList.add("errorClass");
```

```
}
else {
if (document.getElementById("pass1").value.length <= 8) {
printError("passError1", "password length is less than 8");
document.getElementById("pass1").classList.add("errorClass");
} else {
printNoError("passError1", "");
document.getElementById("pass1").classList.remove("errorClass");
}
}

if (document.getElementById("pass2").value.trim() == "") {
printError("passError2", "Please enter confirm password");
document.getElementById("pass2").classList.add("errorClass");
} else {
if (document.getElementById("pass2").value.length <= 8) {
printError("passError2", "password length is less than 8");
document.getElementById("pass1").classList.add("errorClass");
} else if (document.getElementById("pass1").value.trim() !=
document.getElementById("pass2").value.trim()) {
printError("passError2", "Password does not match");
document.getElementById("pass2").classList.add("errorClass");
} else {
printNoError("passError2", "");
document.getElementById("pass2").classList.remove("errorClass");
}
}

if (document.form1.gender.value.trim() == "") {
printError("genderErr", "Please enter gender");
document.getElementById("gender").classList.add("errorClass");
} else {
printNoError("genderErr", "");
document.getElementById("gender").classList.remove("errorClass");
}
if (document.form1.specialization.value.trim() == "") {
printError("specErr", "Please enter specialization");
document.getElementById("specialization").classList.add("errorClass");
} else {
printNoError("specErr", "");
document.getElementById("specialization").classList.remove("errorClass");
}
var email=  document.getElementById("email").value;
if(email !==""){
$.ajax({
url   : "/dis/emailcheck.php",
type  : "POST",
async : false,
data  : {
"emailId" : email
},
success: function(data)
{
if(data==-1){
printNoError("emailError", "");
```

```
document.getElementById("email").classList.remove("errorClass");
} else
{
printError("emailError", "Email Exists already");
document.getElementById("email").classList.add("errorClass");
}
}
});
}

if (document.form1.pi.value.trim() == "" ||document.form1.fn.value.trim() == "" ||
document.form1.ln.value.trim() == "" || document.form1.email.value.trim() == "" ||
document.form1.gender.value.trim() == "" || document.getElementById("pass2").value.trim() == "" ||
document.getElementById("pass1").value.trim() == "" ||
document.getElementById("pass1").value.trim() != document.getElementById("pass2").value.trim() ||
document.form1.specialization.value.trim() == "" ||
document.getElementById("emailError").innerHTML === "Email Exists already") {
return false;
} else {
return true;
}
}

function printError(elemId, hintMsg) {
var elem = document.getElementById(elemId);
elem.classList.add("error");
elem.innerHTML = hintMsg;

}

function printNoError(elemId, hintMsg) {
var elem = document.getElementById(elemId);
elem.classList.remove("error");
elem.innerHTML = hintMsg;
}



function triggerClick() {
document.querySelector('#pi').click();
}
function displayImage(e) {
if (e.files[0]) {
var reader = new FileReader();

reader.onload = function (e) {
document.querySelector('#pd').setAttribute('src', e.target.result);
}
reader.readAsDataURL(e.files[0]);
}
}
</script>
<?php
```

```
if(isset($_POST['submit'])){
$image=basename($_FILES["pi"]["name"]);
$targetDir="profs/";
$targetFilePath=$targetDir.$image;
move_uploaded_file($_FILES["pi"]["tmp_name"],$targetFilePath);

$email=$_POST['email'];
$fn=$_POST['fn'];
$ln=$_POST['ln'];
$gen=$_POST['gender'];
$spec=$_POST['spec'];
$pass=$_POST['pass'];
$sql=mysqli_query($con,"INSERT into login ( `email`, `pass`, `role`, `sta`)
values('$email','$pass','doc',1)");
$roleid = mysqli_insert_id($con);
$sql1="INSERT into d_reg (FirstName, LastName, gender, speciali, Sta, role,
loginid,img,experience,education,phonenumber, address) values
('$fn','$ln','$gen','$spec',1,'doc','$roleid','$image',0,'',0, '')";
echo $sql,$sql1;
if(mysqli_query($con,$sql1)){
echo"<script>location='login.php'</script>";
}else{
echo"<script>location='doctorregister.php'</script>";
}

}
?>
<?php include'footer.php'; ?>
</body>

</html>
```

Login.css

```
body {
background-color: #F3EBF6;
}
.main {
background-color: #FFFFFF;
width: 400px;
height: 400px;
margin: 4em auto;
border-radius: 1.5em;
box-shadow: 0px 11px 35px 2px rgba(0, 0, 0, 0.14);
}
.sign {
padding-top: 86px;
color:#16a085;
font-family: 'Ubuntu', sans-serif;
font-weight: bold;
font-size: 23px;
}
.un {
width: 76%;
color: #777;;
```

```
font-weight: 700;
font-size: 14px;
background: rgba(136, 126, 126, 0.04);
padding: 10px 20px;
border: none;
border-radius: 20px;
outline: none;
box-sizing: border-box;
border: 2px solid rgba(0, 0, 0, 0.02);
margin-bottom: 50px;
text-align: center;
margin-bottom: 27px;
}

form.form1 {
padding-top: 10px;
}
form.form2 {
padding-top: 600px;
}

.pass {
width: 76%;
color:  rgb(26, 53, 66);
font-weight: 700;
font-size: 14px;
letter-spacing: 1px;
background: rgba(136, 126, 126, 0.04);
padding: 10px 20px;
border: none;
border-radius: 20px;
outline: none;
box-sizing: border-box;
border: 2px solid rgba(0, 0, 0, 0.02);
margin-bottom: 50px;
text-align: center;
margin-bottom: 27px;
font-family: 'Ubuntu', sans-serif;
}
.un:focus, .pass:focus {
border: 2px solid rgba(0, 0, 0, 0.18) !important;
}
.submit {
cursor: pointer;
border-radius: 5em;
color: #fff;
background: linear-gradient(to right, #16806a, #0dddb4);
border: 0;
padding-left: 40px;
padding-right: 40px;
padding-bottom: 10px;
padding-top: 10px;
font-family: 'Ubuntu', sans-serif;
margin-left: 8%;
font-size: 13px;
```

```
box-shadow: 0 0 20px 1px rgba(0, 0, 0, 0.04);
}
.forgot {
text-shadow: 0px 0px 3px rgba(27, 109, 82, 0.12);
color: #15d6b0;
padding-top: 15px;
}
a {
text-shadow: 0px 0px 3px rgba(117, 117, 117, 0.12);
color: #15d6b0;
text-decoration: none
}

@media (max-width: 600px) {
.main {
border-radius: 0px;
}
}
#main1{
height: 800px;
width: 480px;
}
a.doctorregister {
float: right;
padding-top: 25px;
padding-right: 20px;
font-size: 1.5rem;
}
#pd
{
display: block;
width: 18%;
margin: 5px auto;
border-radius: 50%;
}
```

Cheart.php

```
<?php
include '../config.php';
session_start();
if($_SESSION['UserID']==""){
header("location:../login.php");
}
$uname=$_SESSION['UserID'];
echo $uname;

$query="select * from login l,u_reg d where l.loginid = d.loginid AND l.loginid='$uname'";
```

```
$result = mysqli_query($con ,$query);
$row=mysqli_fetch_array($result);
?>
<!Doctype HTML>
<html>
<head>
<title></title>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.9.0/jquery.min.js"></script>
<link rel="stylesheet" href="css/style.css" type="text/css"/>
<script src="../patient/js/user_form.js"  type="text/javascript"></script>
<link rel="stylesheet" href="../css/style.css">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
    awesome/5.15.4/css/all.min.css">
<link rel="stylesheet" type="text/css" href="../patient/css/basic style.css">
</head>
<body>
<header class="header">
<a href="#" class="logo"> <i class="fas fa-heartbeat"></i> V-care. </a>
<nav class="navbar">
<a  class="icon-a"><?php echo $row['FirstName'] ?></a>
<a href="../logout.php" class="btn" id="loginbtn" href="../login.php">Logout</a>
</nav>
<div id="menu-btn" class="fas fa-bars"></div>
</header>
<div id="mySidenav" class="sidenav">
<a href="index.php" class="icon-a "><i class="fas fa-user-circle"></i>   Profile</a>
<a href="diseasetype.php"class="icon-a"><i class="fas fa-disease"></i>   Health
    checkup</a>
<a href="feedback.php"class="icon-a"><i class="fas fa-plus-circle"></i>   Feedback</a>
</div>
<div id="main">
<div class="head">
<h1>Select your symptom levels!</h1>
</div>
<div class="container ">
<div id="add_here">
</div>
</div>
<div class="container">
<div class="vertical-gap" align="center">
```

```
<button class="btn btn-default btn-primary col-sm-offset-5" onclick="make_string()">Evaluate
    Symptoms!</button>
</div>
<div id="results">
</div>
</div>
<div class="container">
<div id="precaution" style="margin-top: 30px">
</div>
</div>
</body>
</html>
```

User_form.js

```
$(document).ready(function(){
ajax_call();
});

var urlParams = new URLSearchParams(window.location.search);
var myParam = urlParams.get('id');
var symptom_description = [];
function ajax_call(str)
{
$.ajax({
url   : "../patient/symptom_description.php",
type  : "POST",
async : false,
data  : {
'id' :myParam
},
success: function(data)
{
symptom_description = data.split('|');
}
});
$.ajax({
url   : "../patient/user_form.php",
type  : "POST",
async : false,
data  : {
'id' :myParam
```

```
},
success: function(data)
{
//alert(data);
make_array(data);
}
});
}
var sname = [];
var fv = [];
function make_array(str)
{
var ar = str.split('|');
for (var i = 0; i < ar.length; i++) {
var ind = ar[i].indexOf(',');
sname[i] = ar[i].substring(0,ind);
fv[i] = ar[i].substring(ind+1,ar[i].length).split(',');
//alert(sname[i]+" "+fv[i]);
}
show();
}
function show()
{
var tb1 = '<div class=""><table class="table table-responsive table-bordered
      "><thead><tr><th>Symptom </th><th>Value</th><th>Symptom
      Description</th></tr></thead><tbody>';
var tb2 = '</tbody></table></div>';
for (var i = 0; i < sname.length; i++) {
var box = '<tr><td class = "sname">'+sname[i]+' :- </td>';
box = box +'<td><select class=" form-control" id="s'+i+'">'+(i+1)+' '+sname[i];
for (var j = 0; j < fv[i].length; j++) {
var t;
if(fv[i][j] ==="0"){
t = '<option value="'+fv[i][j]+'">'+"No"+'</option>';
}else if(fv[i][j] ==="1"){
t = '<option value="'+fv[i][j]+'">'+"Yes"+'</option>';

}else if(fv[i][j] ==="3"){
t = '<option value="'+fv[i][j]+'">'+"Mild"+'</option>';
}else{
t = '<option value="'+fv[i][j]+'">'+fv[i][j]+'</option>';
}
```

```
box += t;
}
box+='</select></td>';
box += '<td>'+ symptom_description[i] +'</td></tr>';
tb1 = tb1+box;
}
$("#add_here").append(tb1+tb2);
}

var json;
function make_string()
{
$('#results').children().remove();
var str = '';
for (var i = 0; i < sname.length; i++) {
str = str +sname[i]+','+$("#s"+i).val();
if(sname.length-1!=i)
str += '|';
}
console.log(str)
$.ajax({
url   : "../patient/evaluate.php",
type  : "POST",
async : false,
data  : {
'id' :myParam,
'str':str
},
success: function(data)
{
console.log(data)
json = JSON.parse(data);
var len = json.length;
var i = 1;
var str = '';

var tuple1 = '<td><button class="btn btn-primary" onclick="precaution(this)"  id="';
var tuple2 = '">Precautions</button></td>';
var tb1 = '<div class="vertical-gap"><table class="table table-responsive table-bordered
    "><thead><tr><th>Sr No. </th><th>Disease</th><th>Severity(in %)</th><th>Prediction
    Result</th><th>Precautions</th></tr></thead><tbody>';
```

```
var tb2 = '</tbody></table></div>';
for(key in json){

str += '<tr><td>'+i+'</td>'+'<td>'+key+'</td>'+'<td>'+json[key].toFixed(2)+'</td>';
var danger = '<td class="text-danger">High Possibility of '+key+'</td>';
var normal = '<td class="text-success">Low Possibility of '+key+'</td>';
if(json[key] > 50)
str += danger;
else
str += normal;

str += tuple1+key+tuple2+'</tr>';
console.log(key + "==>" + json[key]);
i++;
}
$('#results').append(tb1+str+tb2);
}
})
}
function accuracy()
{
}
function getJSON(){
return json;
}
function precaution(e)
{
$.ajax
({
url : "../patient/map.php",
type : "POST",
data : {
"disease" : e.id,
"type" : 0
},
success : function(data){
$('#precaution').html('<div id ="prec1" class="jumbotron text-justify" style="font-family:Comic Sans
    MS, cursive, sans-serif">');
$('#prec1').html('<h2>Precaution for '+e.id+' </h2>'+data+'</div>');
$('html, body').animate({
scrollTop: $("#prec1").offset().top
```

```
}, 3000);

},
error : function(err)
{
alert('error in ajax');
}
});
}
```

Map.php

```
<?php
require "../config.php";
$disease = $_POST['disease'];
$type = json_decode($_POST['type']);
$q1 = "select * from disease where dname ='".$disease."'";
$res = mysqli_query($con,$q1);
if($type == 1){
while($row = $res->fetch_array())
{
echo $row['specialist'];
}
}
else
{
while($row = $res->fetch_array())
{
echo $row['precaution'];
}
}
?>
```

Symptom_description.php

```
<?php
require "../config.php";
$did =json_decode($_POST['id']);
$str = '';
$query = "SELECT * FROM mapping WHERE did= '$did' ";
$q_run=mysqli_query($con,$query);
if(mysqli_num_rows($q_run) > 0){
while ($row = mysqli_fetch_assoc($q_run)) {
```

```
$s_id = $row['s_id'];
$q1 = "select  Description from symptom where s_id= '$s_id' ";
$res = mysqli_query($con,$q1);
$row2 = mysqli_fetch_assoc($res);
$str = $str.$row2['Description'].'|';
}
}

echo $str;
?>
```

User_form.php

```
<?php
require "../config.php";
$did =json_decode($_POST['id']);
$all_syms = array_of_syms($con,$did);
function array_of_syms($con,$did)
{
$str = '';
$query = "SELECT s_id FROM mapping WHERE did='$did' ";

$qu_run=mysqli_query($con,$query);
if(mysqli_num_rows($qu_run) > 0){
while ($row = mysqli_fetch_assoc($qu_run)) {
$s_id = $row['s_id'];
$q1 = "SELECT * from symptom where s_id='$s_id'";
$res = mysqli_query($con,$q1);
$row2 = mysqli_fetch_assoc($res);
$id = $row2['s_id'];
$str = $str.$row2['s_name'].','.$row2['fuzzy_set'].'|';
}
}
echo substr($str,0,strlen($str)-1);
}
?>
```

Listdis.php

```
<?php
include '../config.php';
session_start();
```

```
if($_SESSION['UserID']==""){
header("location:../login.php");
}
$uname=$_SESSION['UserID'];
echo $uname;

$query="select * from login l,d_reg d where l.loginid = d.loginid AND l.loginid='$uname'";
//$query = “select * from tbl_login";
$result = mysqli_query($con ,$query);

$row=mysqli_fetch_array($result);
// $imageURL='image/'.$row["image"];
?>

<!Doctype HTML>
<html>
<head>
<title></title>
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
      awesome/5.15.4/css/all.min.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.9.0/jquery.min.js"></script>
<link rel="stylesheet" href="css/style.css" type="text/css"/>
<link rel="stylesheet" href="../css/style.css">

<script type="text/javascript">

var no=0;
$(document).ready(function(){
ajax_call();
});
function ajax_call()
{
$.ajax({
url   : "../doctor/doctor_list.php",
type  : "POST",
async : false,
success: function(data)
{
//console.log(data);
$('#add_here').append(data);
}
```

```
});
}
function disease_info(str)
{
$.ajax({
url   : "../doctor/disease_info.php",
type  : "POST",
async : false,
data  : {
"disease":str,
},
success: function(data)
{
format(data);
}
});
}

function disease_data(e)
{
disease_info(e.id);
}
function delete_dis(did){
if (confirm("Are you sure you want to delete the disease")) {
$.ajax({
url   : "../doctor/del_disease.php",
type  : "POST",
async : false,
data  : {
"diseaseid":did,
},
success: function(data)
{
$('#add_here').children().remove();
ajax_call();
}
});
}
}
function format(str)
{
```

```
var row = '';
var fi = '';
var a = str.split('/');
for(var i=0;i<a.length-1;i++)
{
row = a[i].split('|');
fi += f3(i ,row);

}
$('#disease_info').html(fi);
$('html, body').animate({
scrollTop: $("#disease_info").offset().top
}, 2000);
}

function f3(ind , rows)
{
var eff = ["Yes" , "Maybe" , "No"];
var name = '<div class=" vertical-gap sym_header"><h2>'+(ind+1)+'. '+rows[2]+'</h2><h5> Weight
      of the symptom '+rows[2] + ' = ' +(rows[3])+'</h5></div>';
var tuple1 = '<td><input type="text" class="form-control" disabled value="'
var tuple2 = '"></td><td><input class="form-control input-group-lg" disabled name="cars"  value="'
var tb1 = '<div class="vertical-gap"><table class="table table-responsive table-bordered table-
      hover"><thead><tr><th>Sr No. </th><th>Fuzzy
      Values</th><th>Effect</th></tr></thead><tbody>';
var tb2 = '</tbody></table></div>';

var fuzzy = rows[4].split(',');

var effect = rows[6].split(',');
var s = '';
for(var i=0; i<fuzzy.length;i++)
{
s += '<tr><td>'+(parseInt(i)+1)+'</td>'+tuple1 + fuzzy[i]  + tuple2 + eff[parseInt(effect[i])-
      1]+'"></td></tr>';
}
var tp = (name+tb1+s+tb2);
//alert(tp);
return tp;
}

  </script>
```

```
</head>
<body>
<header class="header">
<a href="#" class="logo"> <i class="fas fa-heartbeat"></i> V-care. </a>
<nav class="navbar">
<a  class="icon-a" style="color: #16a085;">  <?php echo $row['FirstName'] ?></a>
<a href="../logout.php" class="btn" id="loginbtn" href="../login.php">Logout</a>
</nav>
<div id="menu-btn" class="fas fa-bars"></div>

</header>
<div id="mySidenav" class="sidenav">
<a href="index.php" class="icon-a "><i class="fas fa-user-circle"></i>   Profile</a>
<a href="listdis.php"class="icon-a active"><i class="fas fa-disease"></i>   Diseases</a>
<a href="dis.php"class="icon-a"><i class="fas fa-plus-circle"></i>   Add Diseases</a>
</div>
<div id="main">
<div align='right'>
<a href="dis.php" class="btn btn-default btn-primary button">Add Diseases</a>
<a href="listdis.php"  class="btn btn-default btn-primary button">Back</a>
</div>
<div class="container" id="add_here">
</div>
<div class="container" id="disease_info">
</div>
</div>
</body>
</html>
```

Doctor_list.php

```
<?php
require "../config.php";
$all_disease = array_of_disease($con);
function array_of_disease($con)
{
$i = 1;
$output = '<div><table class="table  table-bordered  ">
<tr>
<th >Diseases in database</th>
<th>Actions</th>
```

```
</tr>';

$q1 = "Select * from disease";
$res = mysqli_query($con,$q1);

while($row = $res->fetch_array())
{
$id = $row['did'];
$output .= '<tr><td id='.$row['dname'].'>'.$row['dname'].'</td>
<td><button class="btn btn-default btn-primary button" onclick="disease_data(this)"
    id='.$row['dname'].'>View</button>
<a href="editdis.php?id='.$id.'" class="btn btn-default btn-primary button">Edit</a>
<button type="submit" id="delete" name="submit" class="btn btn-default btn-primary button"
    onclick="delete_dis('.$row['did'].')">Delete</button>
</td>';
$i += 1;
}

echo $output;
}
```

Disease_info.php

```
<?php
require '../config.php';
$disease = $_POST['disease'];
$q1 = "select did from disease where dname ='".$disease."'";
$res = mysqli_query($con,$q1);
$no = mysqli_num_rows($res);
$did = '';
while($row = $res->fetch_array())
{
$did = $row['did'];
}
$q1 = "select did,s.s_id,s_name,weight,fuzzy_set,rangevalue,fv from mapping as m, symptom as s
    where s.s_id = m.s_id and did ='".$did."'";
$res = mysqli_query($con,$q1);
$no = mysqli_num_rows($res);
$str = '';
while($row = $res->fetch_array())
{
$str .=
```

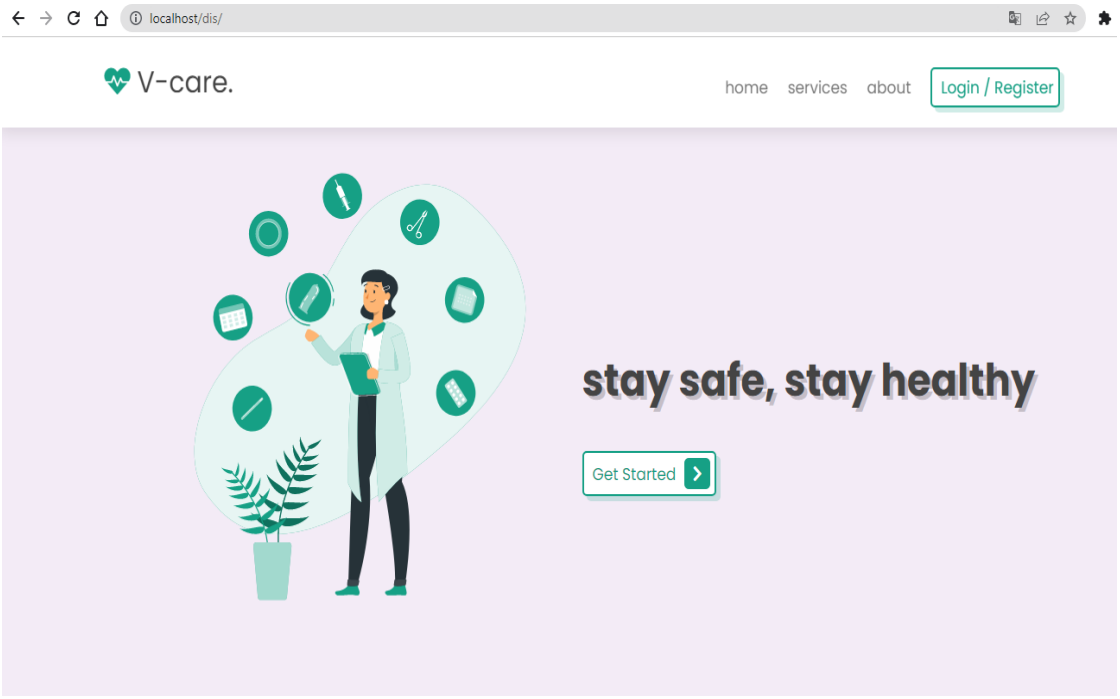
```
        $row['did']."|".$row['s_id']."|".$row['s_name']."|".$row['weight']."|".$row['fuzzy_set']."|".$row['ran
        gevalue']."|".$row['fv']./';
}
echo $str;
?>
```

Del_diesase.php

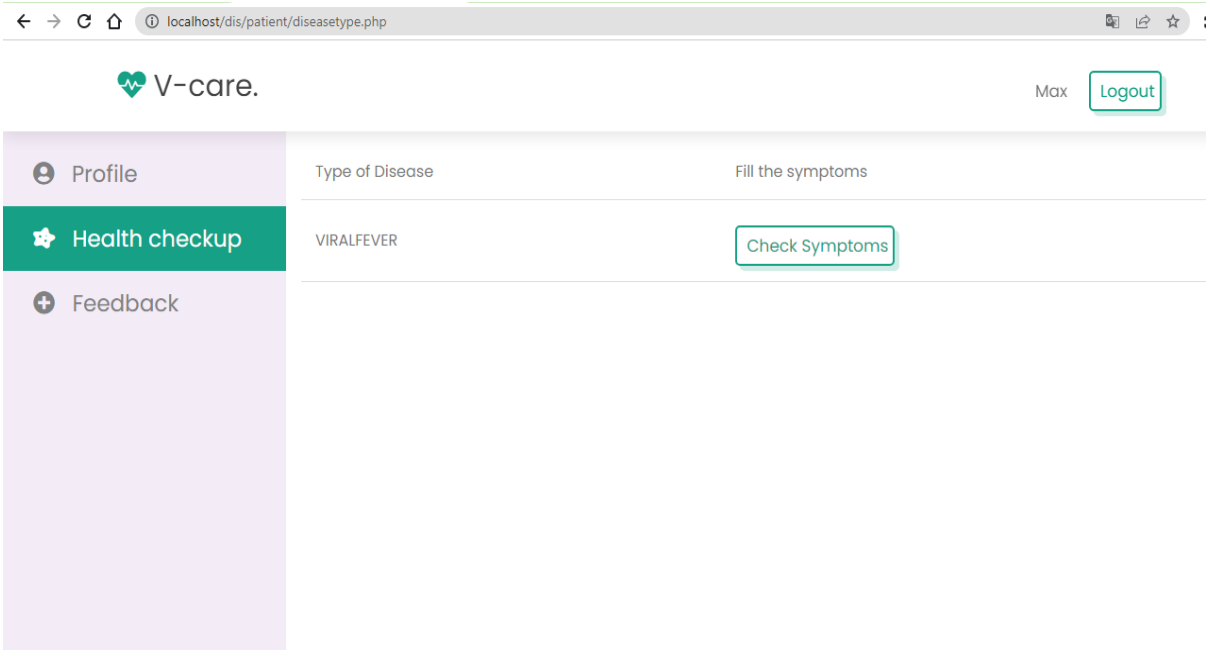
```
<?php
require '../config.php';
$did= $_POST['diseaseid'];
$q1 = "delete from disease where did ='". $did."'";

if (mysqli_query($con, $q1)) {
echo "Record deleted successfully";
} else {
echo "Error deleting record: " . mysqli_error($conn);
}

$query = "SELECT * FROM mapping WHERE did= '$did' ";
$q_run=mysqli_query($con,$query);
if(mysqli_num_rows($q_run) > 0){
while ($row = mysqli_fetch_assoc($q_run)) {
$s_id = $row['s_id'];
$q3 = "Delete from symptom where s_id='$s_id'";

if (mysqli_query($con, $q3)) {
$q4 = "Delete from mapping where s_id='$s_id' and did ='$did'";
if (mysqli_query($con, $q4)) {
echo "Record deleted successfully";
} else {
echo "Error deleting record: " . mysqli_error($conn);
}
}
}
}
?>
```

9.2 Screen Shots

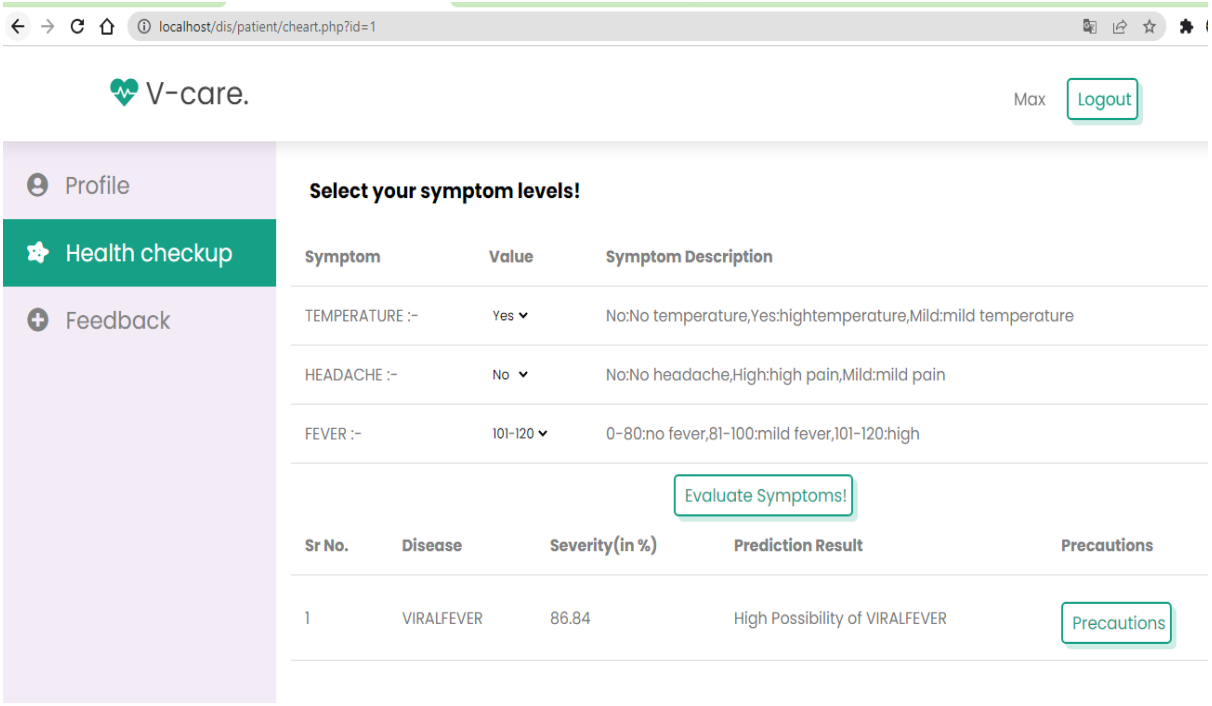
Home page



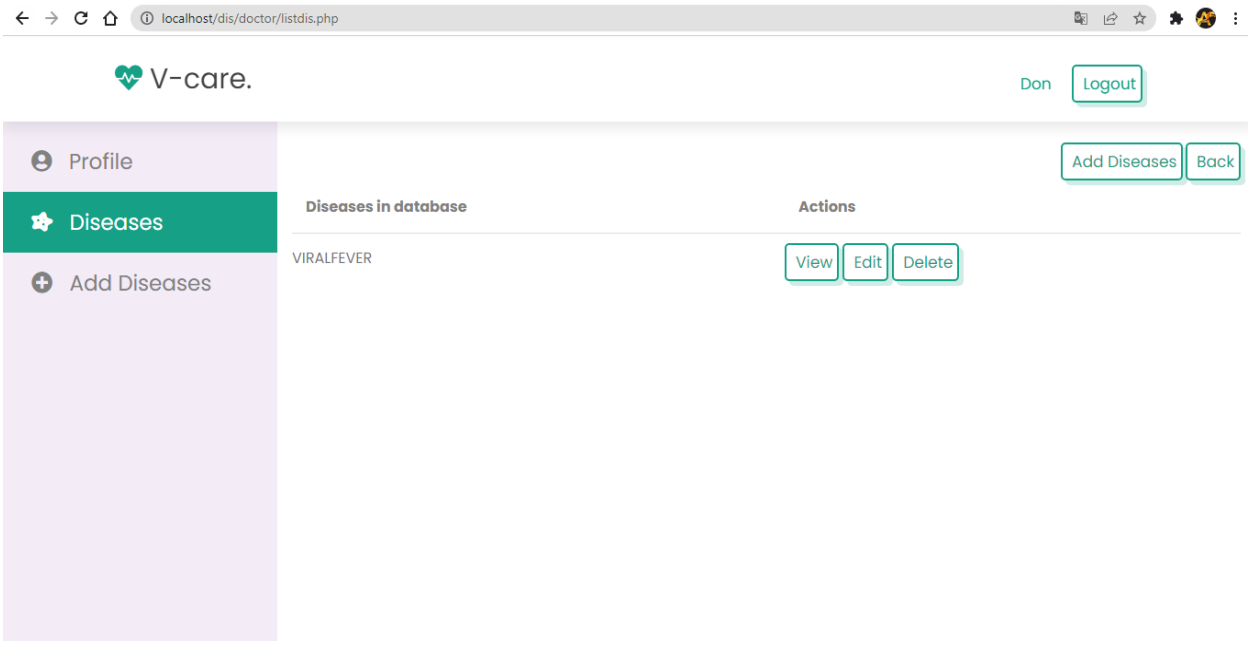
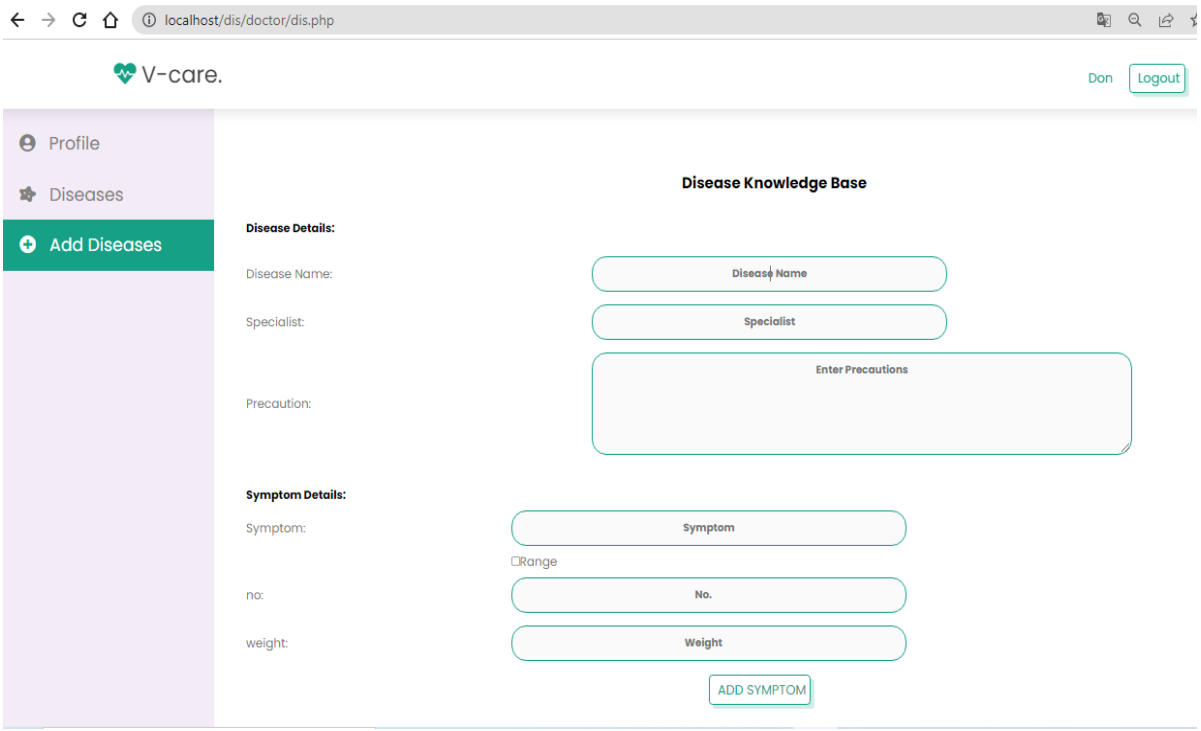
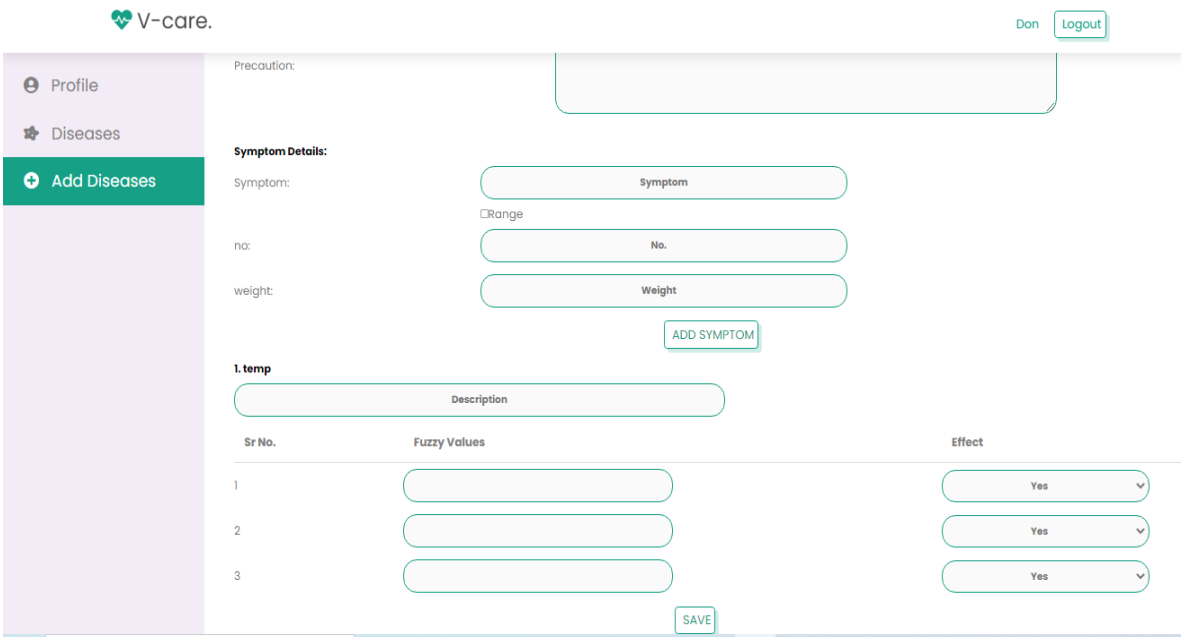
Health Check up Page



Evaluate Disease page



Disease List Page



Add disease page