CHAPTER - 1

1.1 INTRODUCTION

The Disease Prediction and Doctor Recommendation (DPDR) is a medical application that aids patients in predicting diseases and suggesting doctors. This app has been developed with the purpose of providing assistance to patients in selecting the right doctor for the successful prediction of various illnesses. This approach involves collecting and analysing vast the application includes comprehensive information about the doctor, encompassing their qualifications, level of experience, and feedback from previous patients. This serves as a valuable resource for patients seeking additional information. This information can help the user make an informed decision regarding which doctor to contact. By understanding the details provided, the user can decide on the most appropriate doctor to address their specific needs

1.2 Scope of Capstone project

Problem Statement:

There is no physical check-up: Prior to making an appointment with the doctor for a physical examination, there won't be any medical testing. No Face-to-Face interaction to discuss about the symptoms or Disease.

Objectives:

User will get disease information any time.

Reduce the time of user.

User will get Doctor recommendations easily.

User will get disease information anywhere.

Capstone project description:

This project's goal is to establish an online disease prediction and recommendation platform. In this project, doctors must register themselves before customers may visit this portal, where the system predicts the disease and recommends available doctors based on their symptoms. The user will be able to locate the finest doctor for his situation. We will use full stack development languages such as HTML, CSS, PHP, JavaScript, Bootstrap, and MySQL DB to complete this project.

Capstone project Deliverables:

- Admin Modules: -

  Admin Login: Using user name and password admin can login to this project.

  Manage Symptoms & Diseases: Admin will manage the symptoms & Disease.

  Manage Doctor: Admin will recommend the doctors based on users' Disease.

- Doctor Modules: -

  Doctor Login: Using username & Password Doctor will Login to this System.

  View Appointments: The appointments that are scheduled for the day will be visible to the doctor.

- User Modules: -

  Check Symptoms & Disease: In this module user can view the Disease.

  Predict Disease: In this module user can be able to predict Disease.

  View Doctors: In this module user can view the Doctors based on their Disease.

  Book Appointment: This module will allow users to schedule appointments with the doctors of their choice.

Key milestones:

- Admin Modules

Admin Login

Manage symptoms &diseases

Manage Doctor

- Doctor Modules

Doctor Login

View Appointments

- User Modules

Check Symptoms & disease

Predict Disease

View Doctors

Book Appointment

Constraints:

The diseases that will emerge will be unpredictable in our system.

Estimated Capstone project Duration:

The Working Hours this project has taken are 473hrs.

Estimated Capstone project cost:

The total estimated cost of this project ₹48,609/-

CHAPTER - 2

2.1 CAPSTONE PROJECT PLANNING

2.1.1 Work Break-Down Structure

In this project there are three main modules Admin, User & Doctor.

- Admin Modules: -

  Admin Login: Using user name and password admin can login to this project.

  Manage symptoms & Diseases: Admin will manage the symptoms & Disease

  Manage Doctor: Admin will recommend the doctors based on users' Disease.

- Doctor Modules: -

  Doctor Login: Using username & Password Doctor will Login to this System.

  View Appointments: The appointments that are scheduled for the day will be visible to the doctor.

- User Modules: -

  Check Symptoms & Disease: In this module user can view the Disease.

  Predict Disease: In this module user can be able to predict Disease.

  View Doctors: In this module user can view the Doctors based on their Disease.

  Book appointment: This module will allow users to schedule appointments with the doctors of their choice.

Activities & Task

- Admin login (30hrs)

Task: Using HTML, CSS, and Bootstrap, create an admin login page. Admin should login using the parameters specified for validation. For quick access, the administrator must login with the necessary data.

- Manage symptoms & Diseases (36hrs)

  Task: To create form to add Symptoms & Disease. Diseases must be added according to their Symptoms. New symptoms won't manifest themselves.

- Manage Doctor (56hrs)

  Task: To create form to add Doctors. When supervising doctors, it's important to classify the doctors according to their specialties.

- Doctor Login (46hrs)

  Task: To create a Login form for doctors to register them in. Doctor must login with correct details for easy access.

- View Appointments (44hrs)

  Task: The doctors will be able to view the appointments in this Module. While making an appointment If the database is not properly connected, the appointments won't appear in the view appointments.

- Check Symptoms & disease (35hrs)

  Task: To create form to Check Symptoms and Disease. While testing the diseases the, symptoms must be put correctly. While gathering the document the symptoms must match that disease.

- Predict Disease (58hrs)

  Task: To predict the diseases based on the Symptoms. While predicting the disease symptoms must be correctly listed out.

- View Doctors (42hrs)

  Task: Based on the disease or Symptoms, the doctor is recommended. Doctor must be recommended according to the user’s symptoms & diseases.

- Book appointment (46hrs)

Task: This module will allow users to schedule appointments with the doctors of their choice. While booking Appointment the Patient/user must give correct details.

2.1.2 Timeline Development – Schedule

Activities & Task

- Admin Login (30hrs)

  Task: Using HTML, CSS, and Bootstrap, create an admin login page. Admin should login using the parameters specified for validation. For quick access, the administrator must login with the necessary data.

  Collect requirement gathering and requirement analysis to create admin login page.

  Design Form: Create an executive login form.

  Form validation: Use JavaScript to validate the admin password form.

  Database link: Create a database link with PHP and MySQL.

  Testing Form: To try the admin login form for validation and database connection.

  This whole task is done by Palak Jain.

- Manage Symptoms & Disease (36hrs)

  Task: To create form to add Symptoms & Disease. Diseases must be added according to their Symptoms. New symptoms won't manifest themselves.

  Collect requirement gathering and requirement analysis to create Manage Symptoms &Disease page.

  Design Form: Create a form to collect all the symptoms associated with the various illnesses.

  Form validation will be done based on the individuals' symptoms and illnesses.

  Database Connection: PHP & MySQL will then be linked to this form.

Testing Form: To verify validation and the database connection as well as try the additional symptoms and disease.

This whole task is done by Shreyas Kumbar

- Check Symptoms and Disease (35hrs)
  Task: To create form to Check Symptoms and Disease. While testing the diseases the, symptoms must be put correctly. While gathering the document the symptoms must match that disease.

  Collect requirement gathering and requirement analysis to check Symptoms & Disease

  Design Form: Create a form to check all the symptoms associated with the various disease.

  Form validation will be done based on the individuals' symptoms and disease.

  Database Connection: PHP & MySQL will then be linked to this form.

  Testing Form: To verify validation and the database connection as well as try the additional symptoms and disease.

  This whole task is Handled by Palak Jain.

- Predict Disease (58hrs)
  Task: To predict the diseases based on the Symptoms. While predicting the disease symptoms must be correctly listed out.

  Collect requirement gathering and requirement analysis to Predict Disease.

  Design Form: To design the form based on Symptoms &diseases.

  Form validation: Use JavaScript to validate the Symptoms &Disease.

  Database Connection: PHP & MySQL will then be linked to this form.

  Testing Form: To try the predict Symptoms &diseases form for validation and database connection.

  This whole task is Handled by Puneeth. P

- Manage Doctors (56hrs)

  Task: To create form to add Doctors. When supervising doctors, it's important to classify the doctors according to their specialties.

  Collect requirement gathering and requirement analysis to Manage Doctors

  Design Form: Create an executive login form.

  Form validation: Use JavaScript to validate the admin password form.

  Database Connection: PHP & MySQL will then be linked to this form.

  Testing Form: To try the admin login form for validation and database connection.

  This whole task is Handled by Puneeth. P

- Doctor Login (46hrs)

  Task1: To create a Login form for doctors to register them in.

  Task2: Doctor must login with correct details for easy access.

  Collect requirement gathering and requirement analysis to create Doctor Login Page.

  Design Form: create an executive Doctor Login form.

  Form validation: Use JavaScript to validate the Doctor Login password form.

  Database link: Create a database link with PHP and MySQL.

  Testing Form: To try the Doctor login form for validation and database connection.

  This whole task is handled by Shivtej. Ghorpade

- View Doctors (42hrs)

  Task: Based on the disease or Symptoms, the doctor is recommended. Doctor must be recommended according to the user’s symptoms & diseases.

  Database link: Create a database link with PHP and MySQL.

Testing Form: To try the admin login form for validation and database connection.

This whole task is handled by Shreyas. Kumbar.

- Book Appointment (46hrs)

  Task: This module will allow users to schedule appointments with the doctors of their choice. While booking Appointment the Patient/user must give correct details.

  Collect requirement gathering and requirement analysis to create Book Appointment Page.

  Design Form: create a form for users to book appointment.

  Form validation: Use JavaScript to validate the Book Appointment form.

  Database link: Create a database link with PHP and MySQL.

  Testing Form: To try the Book Appointment form for validation and database connection

  This whole task is handled by Palak Jain

- View Appointment (44hrs)

  Task: The doctors will be able to view the appointments in this Module. While making an appointment If the database is not properly connected, the appointments won't appear in the view appointments.

  Collect requirement gathering and requirement analysis to View Appointments.

  Design Form: create a form for doctors to view appointment.

  Form validation: Use JavaScript to validate the View Appointment form.

  Database link: Create a database link with PHP and MySQL.

  Testing Form: To try the View Appointment form for validation and database connection

This whole task is handled by Shivtej Ghorpade.

2.1.3 Cost Break-down Structure

Analyse your Work Breakdown Structure

- Admin Modules

  Admin Login

  Manage symptoms & diseases

  Manage Doctor

- User Modules

  Check Symptoms & disease

  Predict Disease

  View Doctors

  Book Appointment

- Doctor Modules

  Doctor Login

  View Appointment

Estimate the labour cost of work: -

Module Name	Task Name and Their working hours	Total Working Hours	Salary Paid (in hrs)	Total Salary (in ₹)
Admin Login	• Task1: Create an admin login page using HTML, CSS and Bootstrap.(12hrs) • Task2: Admin should log in according to validation parameters.(10hrs) • Task3: Admin must login with correct details for easy access. (8hrs)	30 hrs	30*70	2100
Manage Symptoms &Diseases	• Task1: Create a form to add symptoms and diseases.(12hrs) • Task2: Diseases must be added based on their symptoms. (11hrs) • Task3: New symptoms will not appear due to a lack of symptoms. (13hrs)	36 hrs	36*70	2520
Check Symptoms &Diseases	• Task1: Create a form to check symptoms and diseases. (10hrs) • Task2: Symptoms must be correctly identified when testing diseases.(11hrs) • Task3: Symptoms must match the disease to be included in the document.(14hrs)	35 hrs	35*70	2450
Predict Diseases	• Task1: Predicting diseases based on symptoms is essential.(27hrs) • Task2: The most important idea is to accurately identify the symptoms of a disease.(31hrs)	58 hrs	58*70	4060

Manage Doctor	• Task1: To create form to add Doctors. (24hrs) • Task2: When supervising doctors, it's important to classify the doctors according to their specialties. (32hrs)	56 hrs	56*70	3920
Doctor Login	• Task1: To create a Login form for doctors to register them in. (23hrs) • Task2: Doctor must login with correct details for easy access. (23hrs)	46 hrs	46*70	3220
View Doctor	• Task1: Based on the disease or Symptoms, the doctor is recommended.(20hrs) • Task2: Doctor must be recommended according to the user's symptoms &diseases. (22hrs)	42 hrs	42*70	2940
Book Appointment	• Task1: This module will allow users to schedule appointments with the doctors of their choice.(24hrs) • Task2: While booking Appointment the Patient/user must give correct details.(22hrs)	46 hrs	46*70	3220
View Appointment	• Task1: The doctors will be able to view the appointments in this Module. (20hrs) • Task2: Database connection issue prevents appointments from appearing in appointment view. (24hrs)	44 hrs	44*70	3080

Table.:2.1.3.1

Estimate the cost of materials:

Cloud Server Cost: ₹5000/-

PHP Designer: $42 (Approx ₹3444/-)

XAMPP Server is open-source software.

Overhead Cost: If a live server is going to cost more, we will let you know.

Build Contingency into your CBS:

We must complete the job in accordance with the projected cost.

Final-Check:

Estimated Cost:	₹48,609/-
Labour Cost:	₹27,510/-
Material Cost:	₹8,444/-
Profit of the Project: ₹12655/-	

Table.:2.1.3.2

2.1.4 Capstone project Risks assessment

Risk occurred in our Project:

Cost risk is the risk of exceeding budget or failing to deliver fair value.

Resource risk occurs when not having enough resources to complete a project.

Time risk affects budget, delivery date, and overall results.

Validation Protocol is defined as a written plan describing the process to be validated, including production equipment and how validation will be conducted.

Requirement specifications are developed, the requirements mentioned in this document are validated. There can be illegal, impractical solution or experts may interpret the requirements incorrectly.

Asking and collecting top-level requirement from all relevant users of the project.

Creating a visual part of the process and diagram to easily design.

The DB must be similar to the DB name used in the code.

When the server is on high use the no data is viewed.

The database name and table name were different while creating the table in the predict illness module.

To ensure that the Manage Doctor Module functions properly, the parameters of the Doctor Login Form must adhere to the specifications.

When validating the doctor login form, there was no update button to update the hospital details.

If the View Doctor Form requirement collecting and analysis change, the entire form design changes as well.

2.2 REQUIREMENT SPECIFICATION

2.2.1 Functional requirements:

- Admin: -

  He or she will enter a username and password to log in.

  Admin will manage illness and symptom management.

  He will make a doctor recommendation depending on your "disease."

  Sign off.

- Doctor: -

  The doctor will first register with all of his information.

  He or she will enter a username and password to log in.

  The doctor will view the appointments that have been set up.

  Log off, doctor.

- User: -

  The user will examine his symptoms.

  Al will assist the user in disease prediction.

  The user may view the doctor based on his ailment.

  The user can make a doctor's appointment.

2.2.2 Non-functional requirements:

- Admin: -

  id

  name

  email

  mobile

  username

  password

- Appointment: -
  id
  date
  name
  phone
  email
  user_id
  disease_predicted

- Disease: -
  id
  disease_name
  cause

- link symptoms: -
  id
  organ_id
  disease_id
  symptoms_id

- Organs: -
  id
  organ_name

- Symptoms: -
  id
  symptom_name

- users: -
  id
  hospital_name
  hospital_phone
  hospital email

hospital_city

hospital_address

doctor_name

doctor_phone

doctor_qualification

doctor_speciality

photo

username

password

status

2.2.3 User Input:

- Admin Login: -

  Enter Username

  Enter Password
- Manage Symptoms and disease

  Add Organs

  Add Symptoms

  Add Disease
- Check Symptoms and Disease

  Selecting organs

  Selecting symptoms
- Predict Disease: -

  Disease is Predicted by system.
- Manage Doctor: -

  Admin accepts the request of hospital/doctor.
- Doctor Login: -

  Enter Username

  Enter Password
- View Doctor: -

  Doctors are recommended based on symptoms and doctor.

- Book Appointment: -

  Patients/user will book appointment by adding correct Date, Time, & their personal details.
- View Appointment: -

  Enter Username.

  Enter Password.

  To view the Appointment.

2.2.4 Technical Constraints:

Requirements for developing: -

Average I3 processor.

Average 4GB RAM.

Average 512GB hard disk.

Hardware requirement for deployment: -

Average I3 processor.

Average 2GB RAM.

Average 256GB storage.

Software requirement for development: -

OS (Window’s, MAC, Linux).

XAMP Server.

VS Code.

Chrome.

Software requirement for deployment: -

OS.

Chrome.

2.3 DESIGN SPECIFICATION

2.3.1 Chosen System Design:

System Architect:

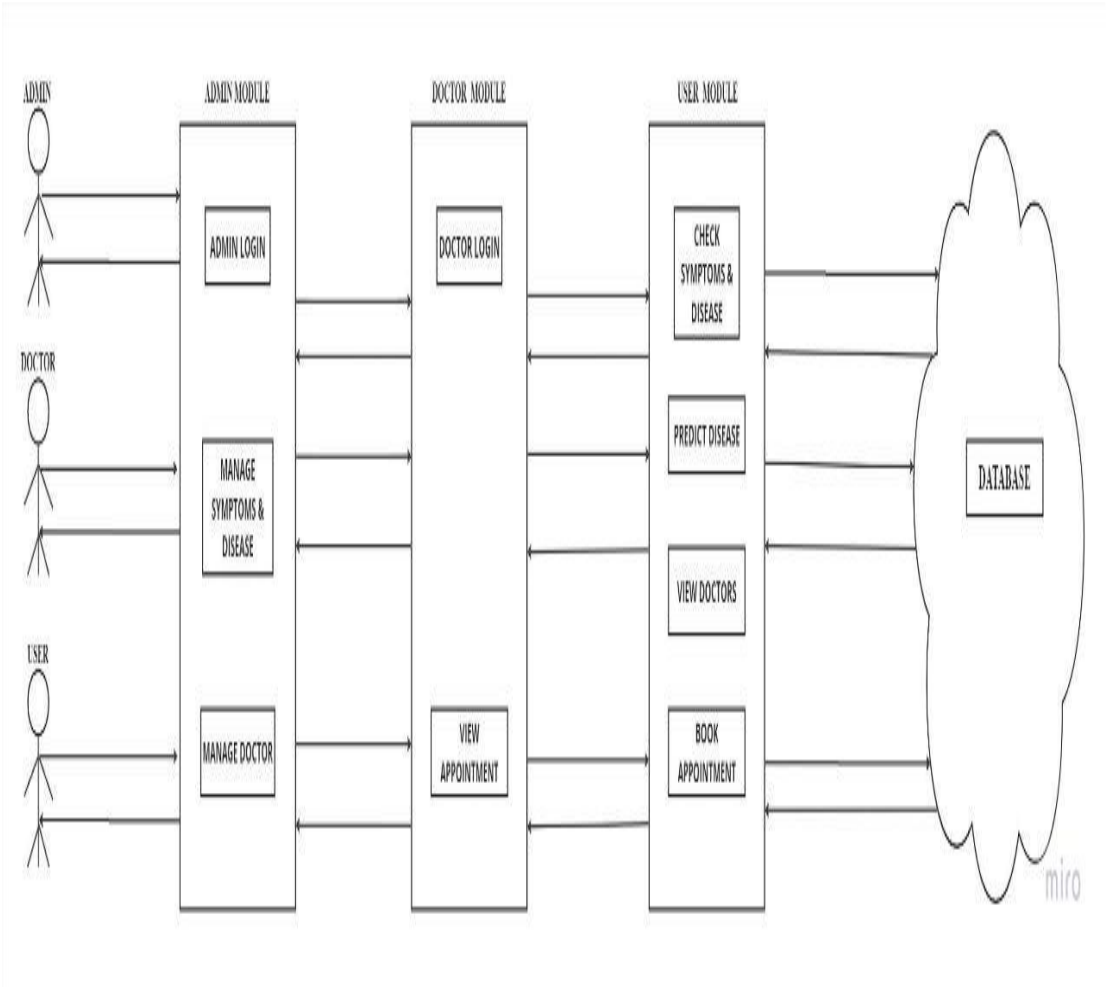

Fig.: 2.3.1.1

System architecture diagrams offer a visual representation of a system's numerous components and demonstrate how they speak to one another and interact. The architecture and structure of a system are described in these diagrams. Identification of requirements that have an impact on the application's structure is the architecture's main objective. How the modules and database communicate with one another is depicted in the figure below. We have three primary modules, including an admin module, a doctor module, and a user/patient module.

2.3.2 Discussion of Alternative Designs:

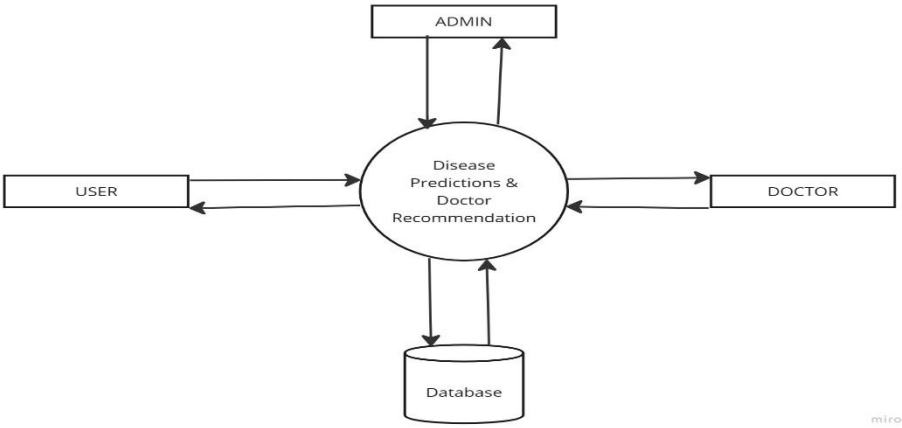

Fig.:2.3.2.1

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination.

Admin: -

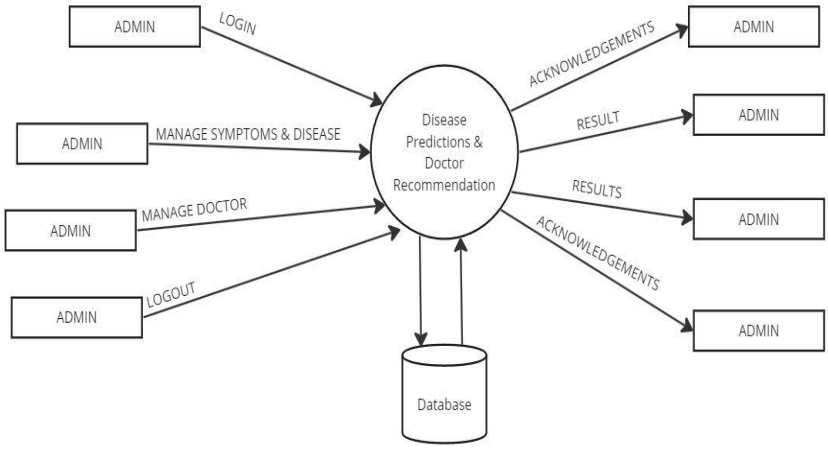

Fig.: 2.3.2.2

A data flow diagram (DFD) is a graphical or visual depiction that describes how data is moved through an organization's activities using a standardised set of symbols and notations. The path that data follows between external entities, processes, and data repositories is known as data flow. Arrows are used to depict

the interaction between the other components, and they are usually labelled with a brief data name, such as "Manage symptoms & Diseases."

Doctor: -

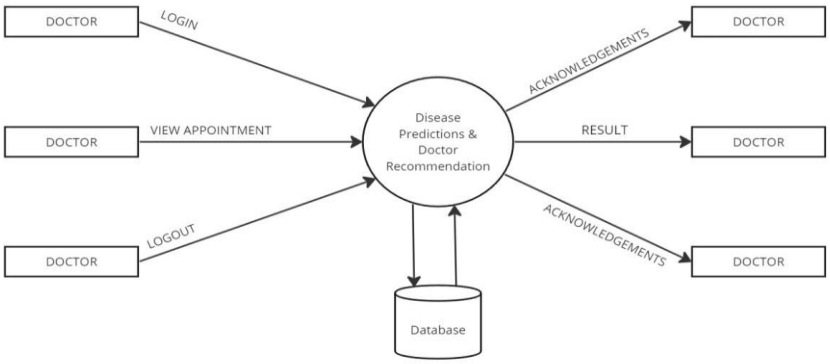

Fig.: 2.3.2.3

A data flow diagram (DFD) is a graphical or visual depiction of a business's processes using a standardised set of symbols and notations. The path that data follows between external entities, processes, and data repositories is known as data flow. It depicts the interaction with the other components and is represented by arrows, which are usually labelled with a brief data name, such as "Doctor Login."

User: -

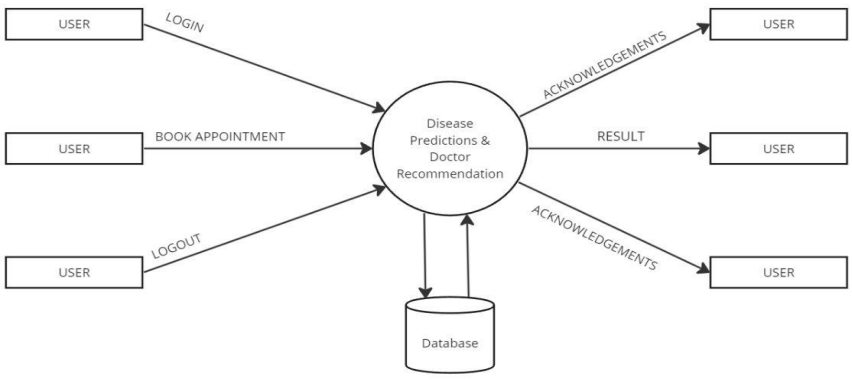

Fig.: 2.3.2.4

A data flow diagram (DFD) is a graphical or visual depiction of a business's activities through data movement that uses a standardised set of symbols and

notations. Data flow is the path that information travels as it moves between external entities, processes, and data repositories. It represents the interface with the other elements and is represented by arrows, which are often labelled with a brief data name, such as "Book Appointment."

2.3.3 Detailed Description of Components /Subsystems:

Component diagram:

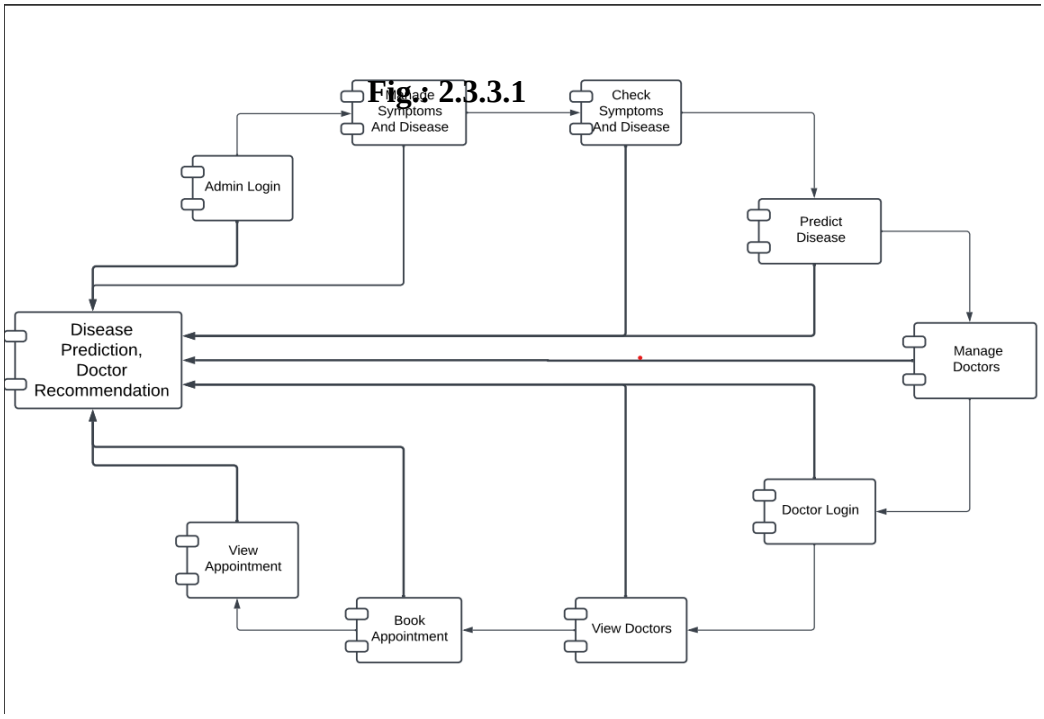

Fig : 2.3.3.1

Fig 2.3.3.1

A component diagram is used to illustrate the connections between various system components. Component diagrams display the organisation of the software system, which includes information on the interfaces and dependencies of the programme components. Here are the elements. Here, we have displayed the module dependencies in this diagram. Admin login controls the symptoms and diseases, while the check symptoms and disease are based on Manage symptoms and disease, Predict disease is dependent on Manage symptoms and disease, Manage doctor is based on Admin, then the doctor login is dependent on Manage doctor, View doctor is dependent on the doctor login, and Book Appointment is based on View Doctors & View Appointment is based on Book Appointment.

2.3.4 Component 1- n

- Admin Login

  We first gathered the prerequisites for admin login, then we analysed the information we had. The admin login form is then created. The page is validated after it has been designed, and each field of the form needs to be filled out correctly. After that, we connected the database. After finishing all the processes, the testing is finished.

- Manage Symptoms & Disease

  For symptom and illness management We initially gathered requirements, then analysed those requirements. Then we built the add symptoms and add disease pages. After constructing both pages, it is validated where the symptoms are defined to fit the disease. Then we connect to a database. Finally, after completing all of the stages, testing is completed.

- Check Symptoms & Disease

  We first gathered the criteria for the do Check Symptoms & disease, analysed those needs, and then designed the form. Following the evaluation of the designed page, we link to the database, and finally all the steps are finished. The testing is over.

- Predict Disease

  In order to predict disease, we first collected the necessary information, which we then analysed. then the form was created. The website is validated when the disease is predicted once it has been designed. then we connect to the database. and lastly, once all the stages have been completed, testing is finished.

- Manage Doctor

  For Managing Physician, we initially gathered the requirements, then we analysed them. then the form for managing the doctors was created. The page is evaluated after it has been designed. then we connect to the database. and lastly, once all the stages have been completed, testing is finished.

- Doctor Login

  We first gathered the criteria for the doctor login, analysed those needs, and then designed the form for the doctor login. Following the evaluation of the designed page, we link to the database, and finally all the steps are finished. The testing is over.

- View Doctor

  For view doctor, we first gathered the requirements, then analysed them, and finally designed the form. Following the evaluation of the designed page, we link to the database, and finally all the steps are finished. The testing is over.

- Book Appointment

  For making an appointment book. collected the needs, examined them, and then created the form that has been. users have scheduled appointments with doctors. Following When all of the stages are complete and the testing is complete, the evaluation of the design page will link to the database.

- View Appointment

  Gather the requirements for seeing an appointment first, analyse them, and then create the form that users will use to view the appointment. Next, doctors Check the appointment on his form by going there. Set the time as he chooses. When all testing stages are complete, the evaluation of the design page will link to the database.

CHAPTER- 3

3.1 APPROACH AND METHODOLOGY

3.1.1 Discuss the Technology:

Web technology

Web technology refers to the means by which computer communicate with each other using markup languages & multimedia package.

Browsers

HTML and CSS

Programming Languages

Frameworks

Web Servers

Databases

Protocols

Data Formats

Advantages:

Lower Development Cost.

Easy installation & Maintenance.

Adaptable to changing Workloads.

Centralized Security.

Increased Efficiency.

Reduced Hardware Costs.

Cloud based technology

cloud computing is a range of services delivered over the internet, or “the cloud.” It means using remote servers to store and access data instead of relying on local hard drives and private datacentres.

Types of Cloud Computing:

SAAS: - Software as a Service.

PAAS: - Platform as a Service.

IAAS: - Infrastructure as a Service.

Advantages:

Faster time to market.

Scalability and flexibility.

Cost savings.

Better collaboration.

Data loss prevention.

Advanced security.

Open-Source web technology

Open-source software (OSS) is software that is distributed with its source code, making it available for use, modification, and distribution with its original rights. Source code is the part of software that most computer users don't ever see; it's the code computer programmers manipulate to control how a program or application behaves.

We do not need to buy any software or libraries for our project.

- Web server, Apache

  Web server, Apache is a open source platform & responsible for accepting directory (HTTP) requests from Internet users and sending them their desired information in the form of files and Web pages.

- XAMPP

  XAMPP is a cross-platform and open-source tool, which makes it an ideal choice of web developers. It is the acronym of X-cross platform, Apache, MySQL, PHP, and Perl.

- PHP

  PHP is an open-source scripting language used for creating dynamic and interactive web pages and various digital platforms.

- PhpMyAdmin

  PhpMyAdmin is an open source and free administration tool for MySQL.

3.1.2 Methodologies:

Agile Methodology is a process for managing a project that involves constant collaboration & working in iteration.

Agile project Management work off the basis that a project can be continuously improved upon thought its life cycle, with changes being made quickly & responsibility.

Phases Of Agile Model: -

Requirement Gathering.

Design the requirement.

Construction/Iterations.

Testing/Quality Assurances.

Deployment.

Feedback.

Agile Diagram:-

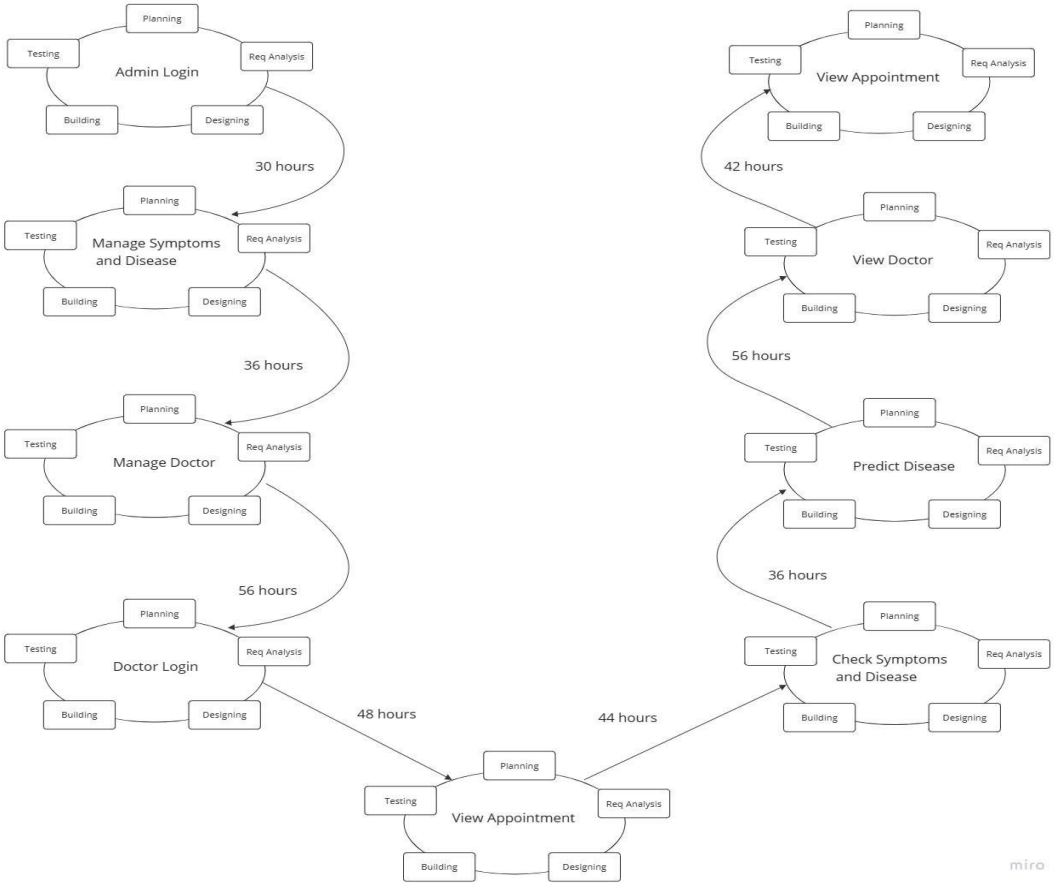

Fig.: 3.1.2.1

3.1.3 Use Cases:

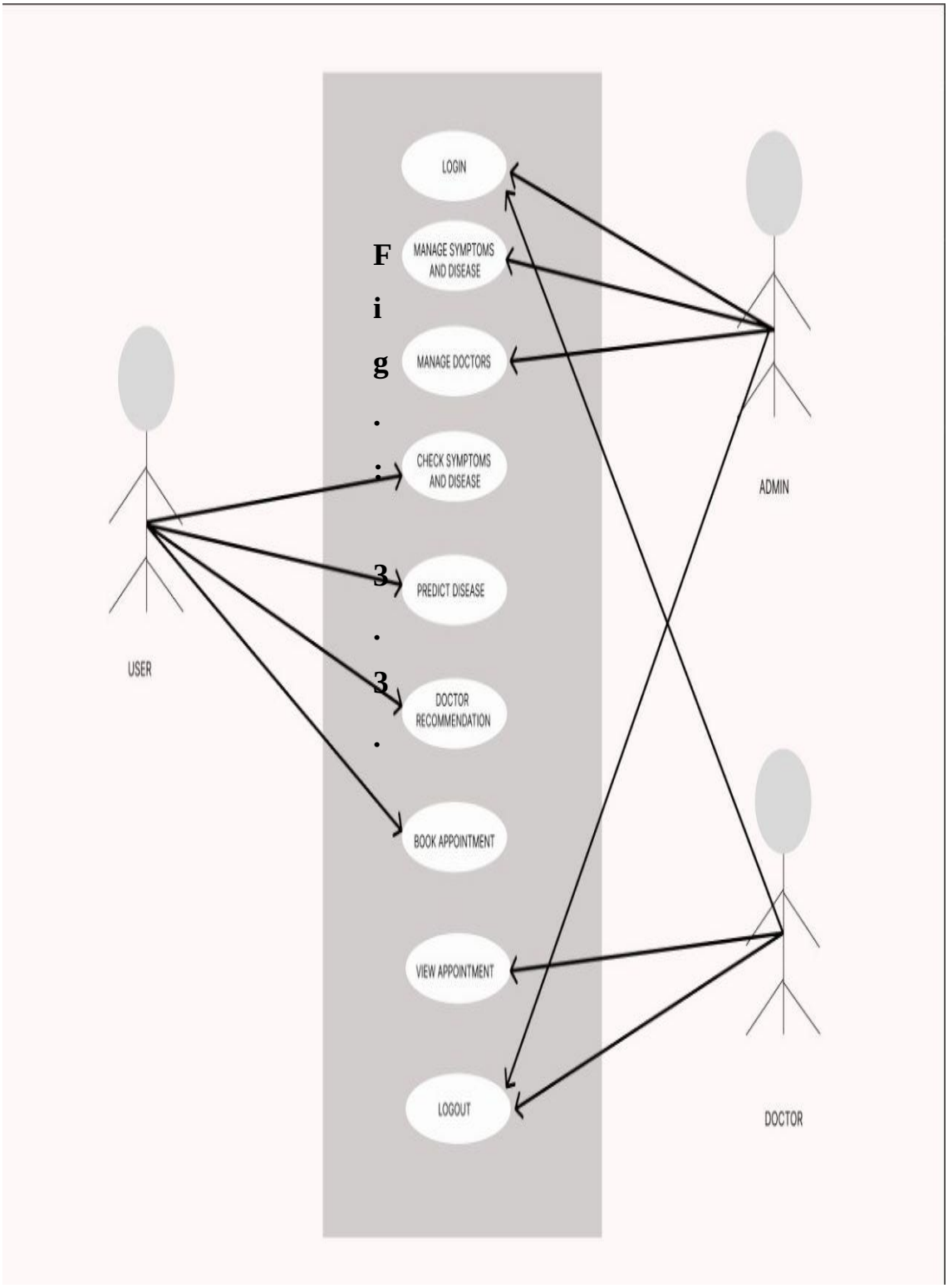

Fig.:3.1.3.1

3.1.4 Programming:

- HTML: -

  HTML stands for Hyper-Text Markup Language. It is used to design web pages using a markup language. HTML is a combination of Hypertext and Markup language. Hypertext defines the link between web pages. A markup language is used to define the text document within the tag which defines the structure of web pages.

- CSS: -

  CSS stands for Cascading Style Sheet.
  CSS is used to design HTML tags.
  CSS is a widely used language on the web.
  HTML, CSS and JavaScript are used for web designing.

- JAVASCRIPT: -

  JavaScript (Js) is a light-weight object-oriented programming language which is used by several websites for scripting the webpages. It is an interpreted, full-fledged programming language that enables dynamic interactivity on websites when applied to an HTML document.

- BOOTSTRAP: -

  Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites. Designed to enable responsive development of mobile-first websites, Bootstrap provides a collection of syntax for template designs.

- PHP: -

  PHP stands for Hypertext Pre-processor.
  PHP is an interpreted language, i.e., there is no need for compilation.
  PHP is faster than other scripting languages, for example, ASP and JSP.
  PHP is a server-side scripting language, which is used to manage the dynamic content of the website.
  PHP can be embedded into HTML.
  PHP is an object-oriented language.

3.1.5 Analysis:

- **Existing system:** Surveys conducted with current patients have shown that if patients want to be evaluated for their illness, they must go to the hospital.

- **Proposed System:-** we are going to develop this is prediction and doctor recommendation the purpose of this project is create Disease Prediction online and recommendation in this project doctor needs to register to this portal and user can visit to this portal and as per user symptoms system predicts the user disease And that time system will recommend the available doctor through this portal User will get perfect doctor for his disease to implement this jack we're going to use full stack development languages like PHP, JavaScript, Bootstrap, MySQL DB, HTML and CSS.

3.1.6 Process Design:

Tables: -

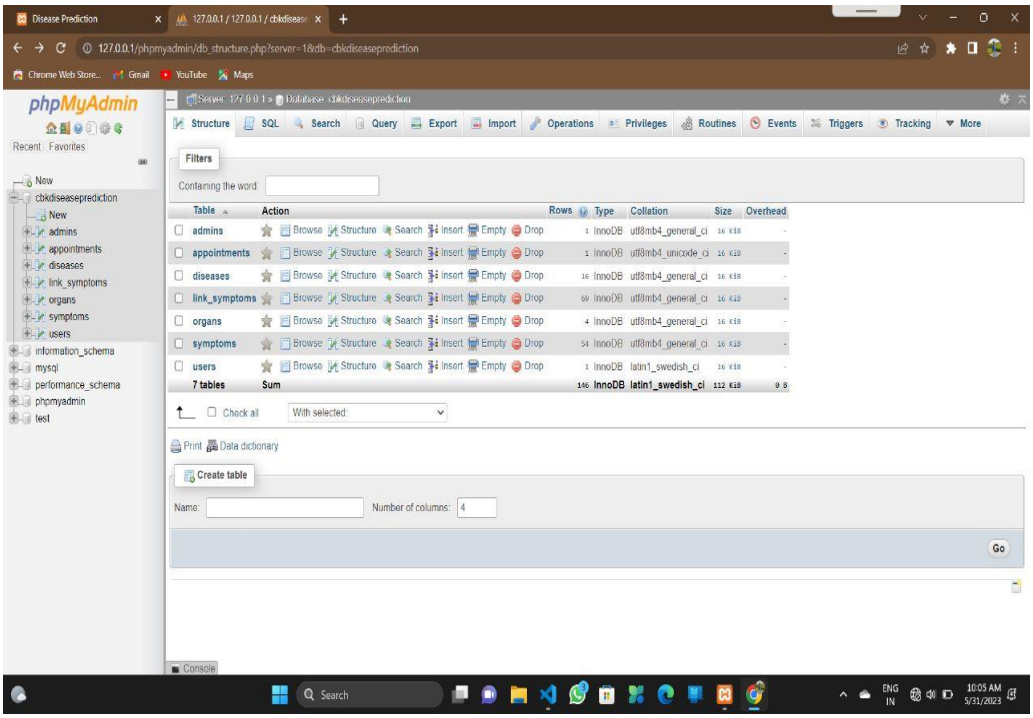

Fig.: 3.1.6.1

Admin:

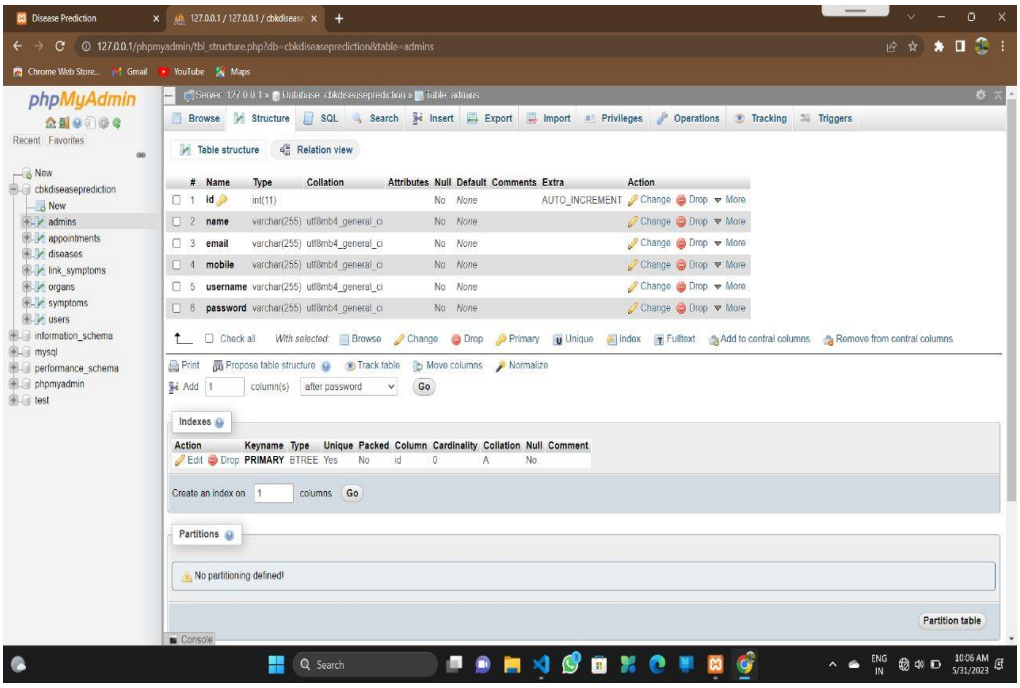

Fig.:3.1.6.2

Appointments: -

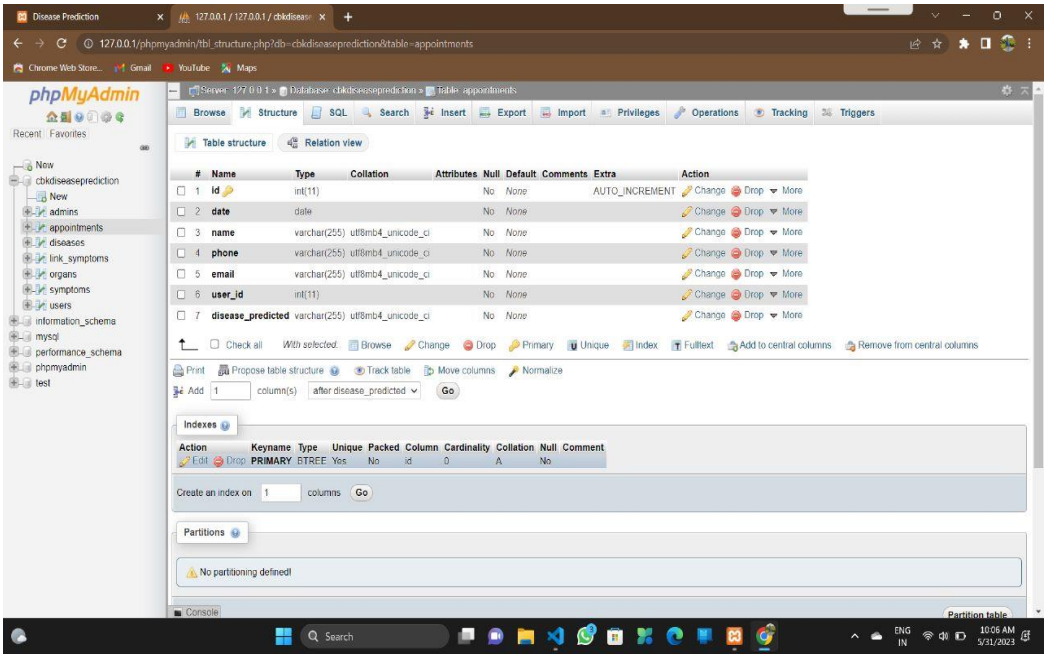

Fig.:3.1.6.3

Disease

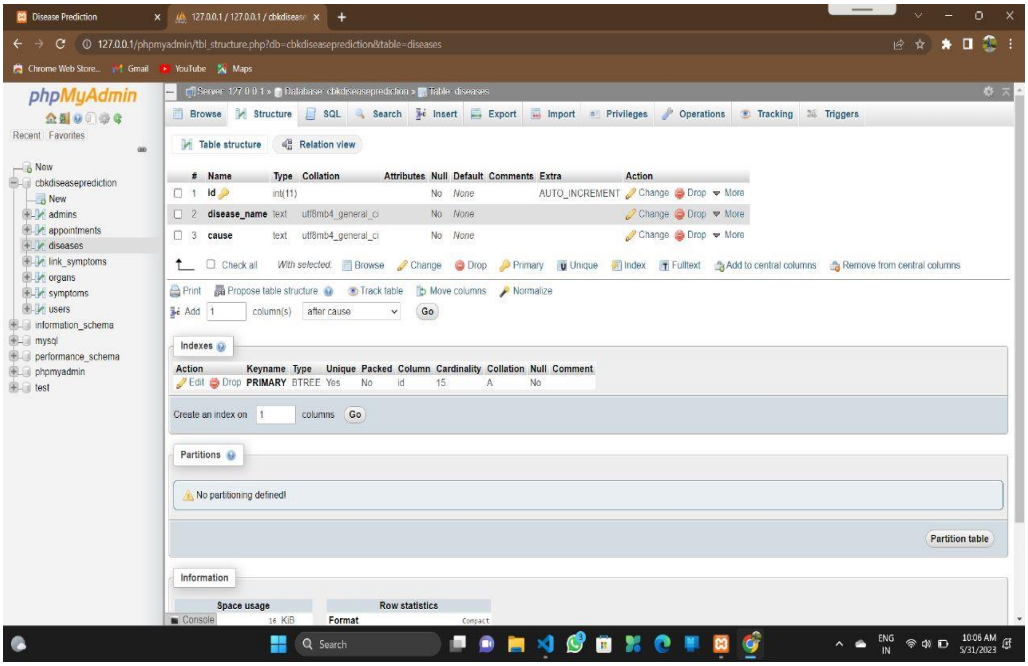

Fig.:3.1.6.4

Link Symptoms:

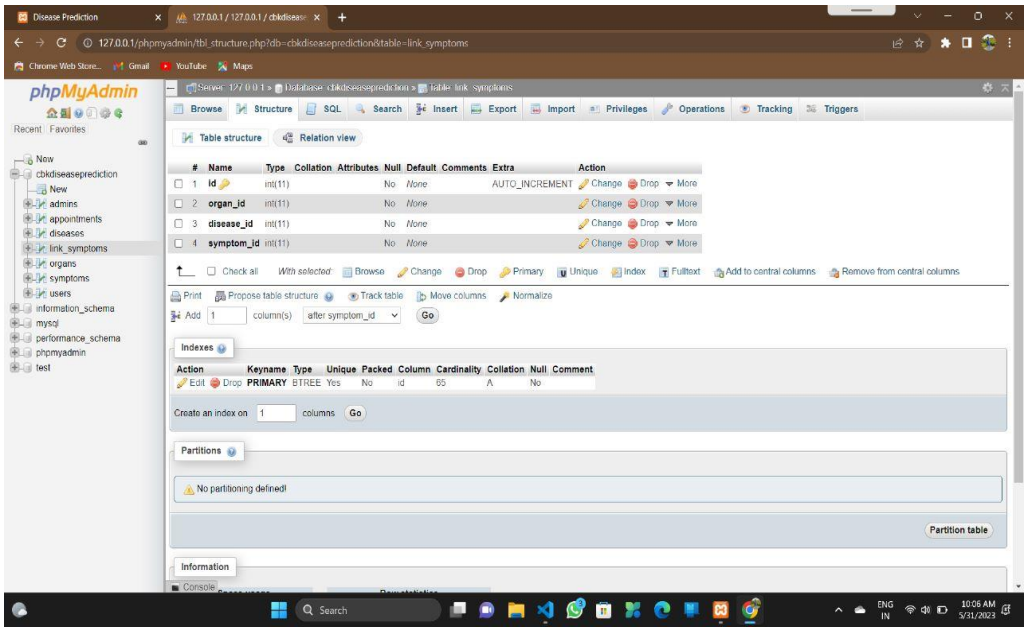

Fig.:3.1.6.5

Organs:

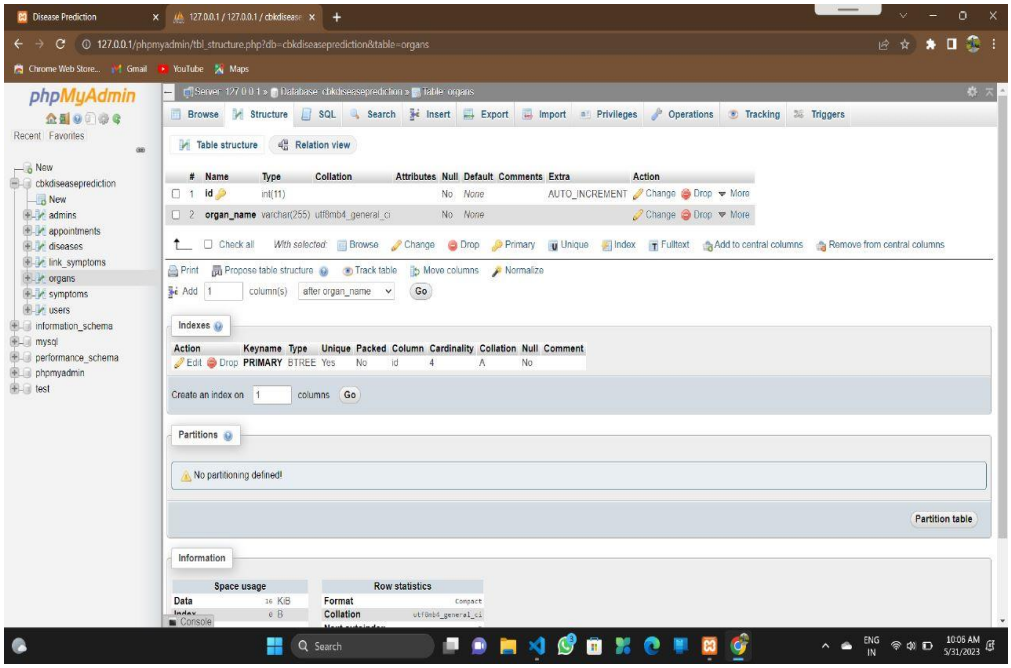

Fig.:3.1.6.6

Symptoms:

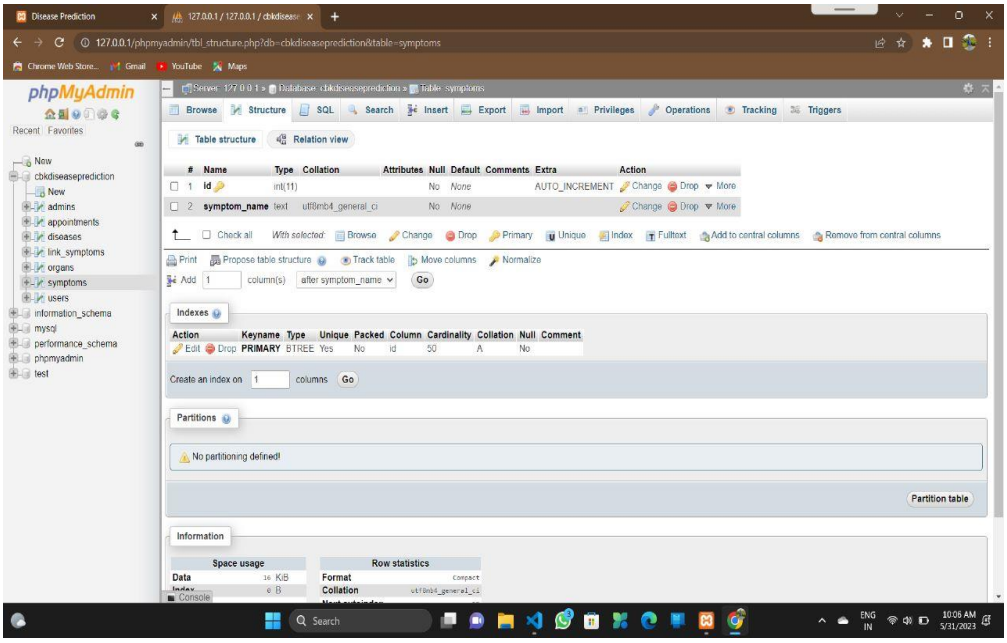

Fig.:3.1.6.7

User:

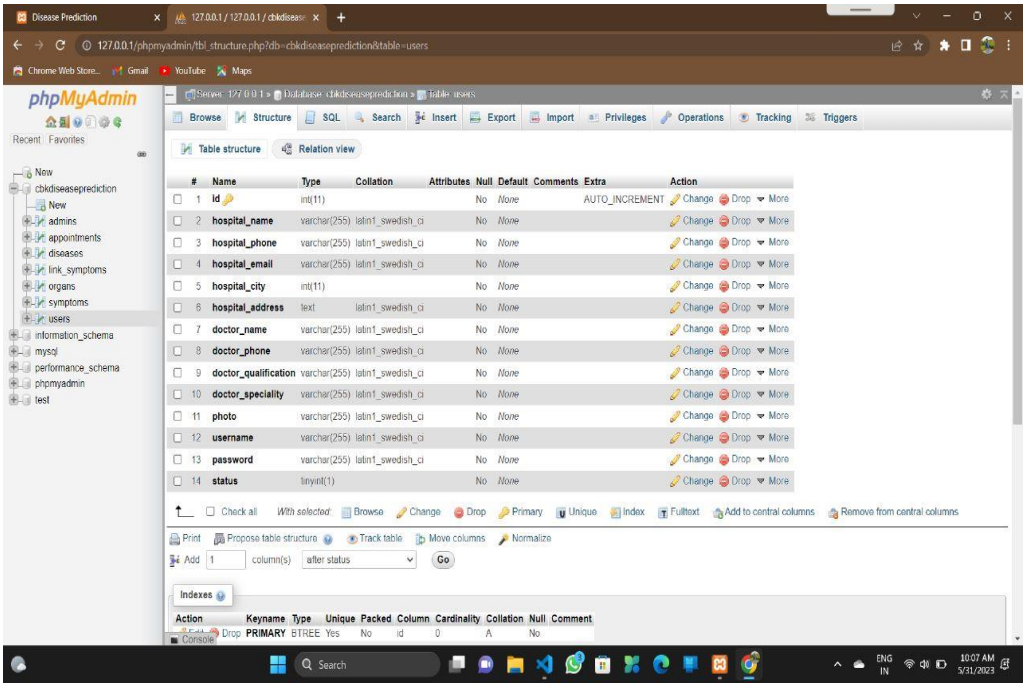

Fig.:3.1.6.8

3.1.7 Product Design:

Dashboard

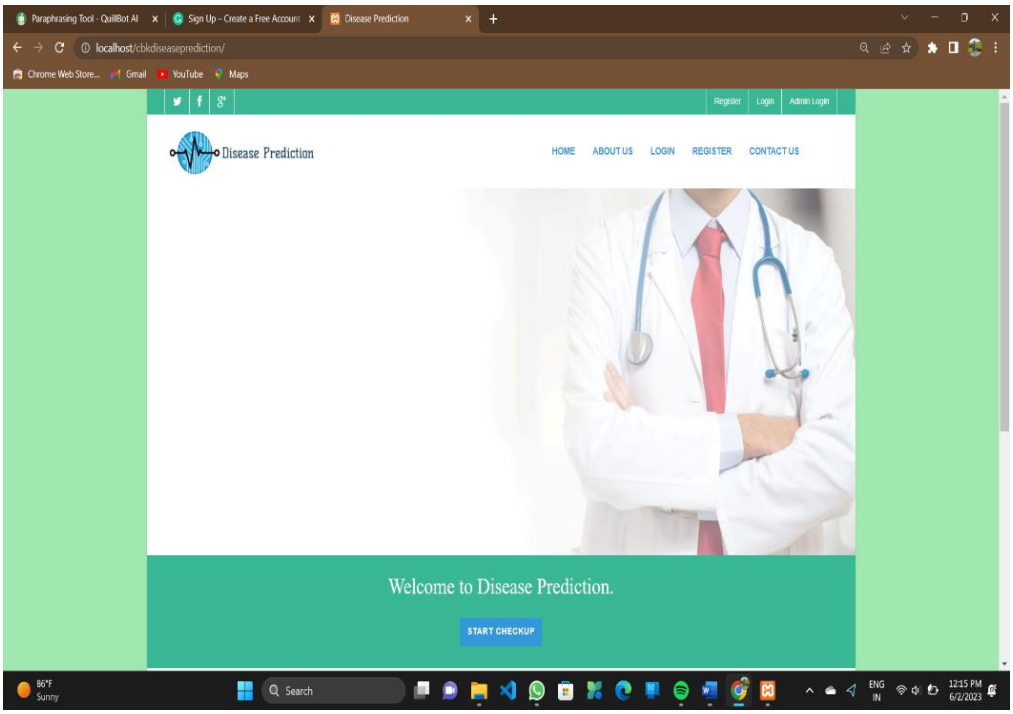

Fig.:3.1.7.1

About us

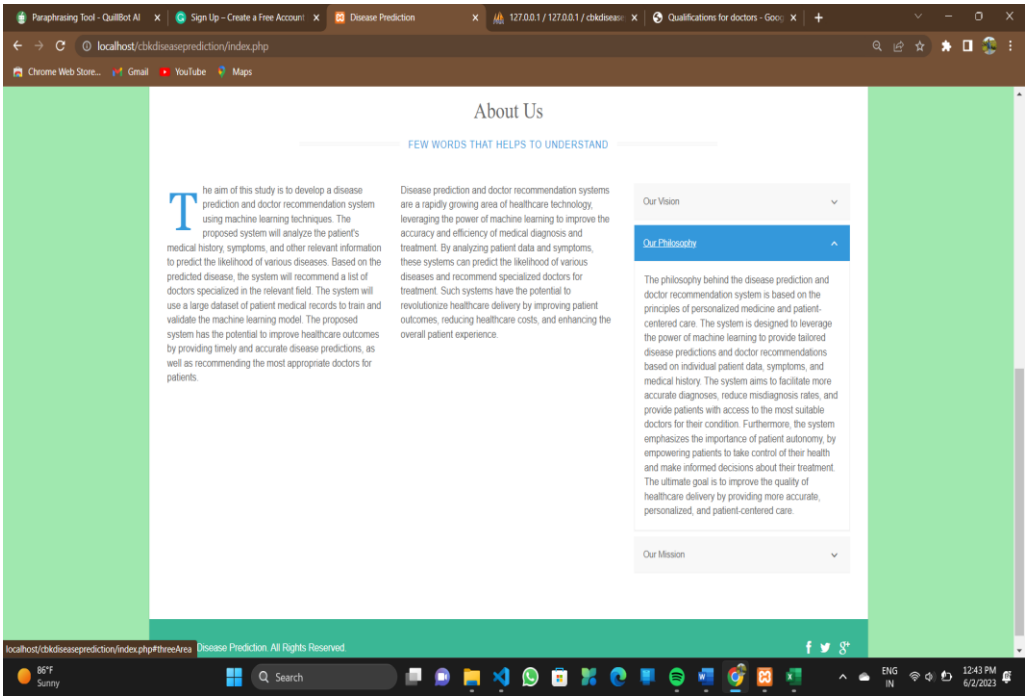

Fig.:3.1.7.2

Admin Login

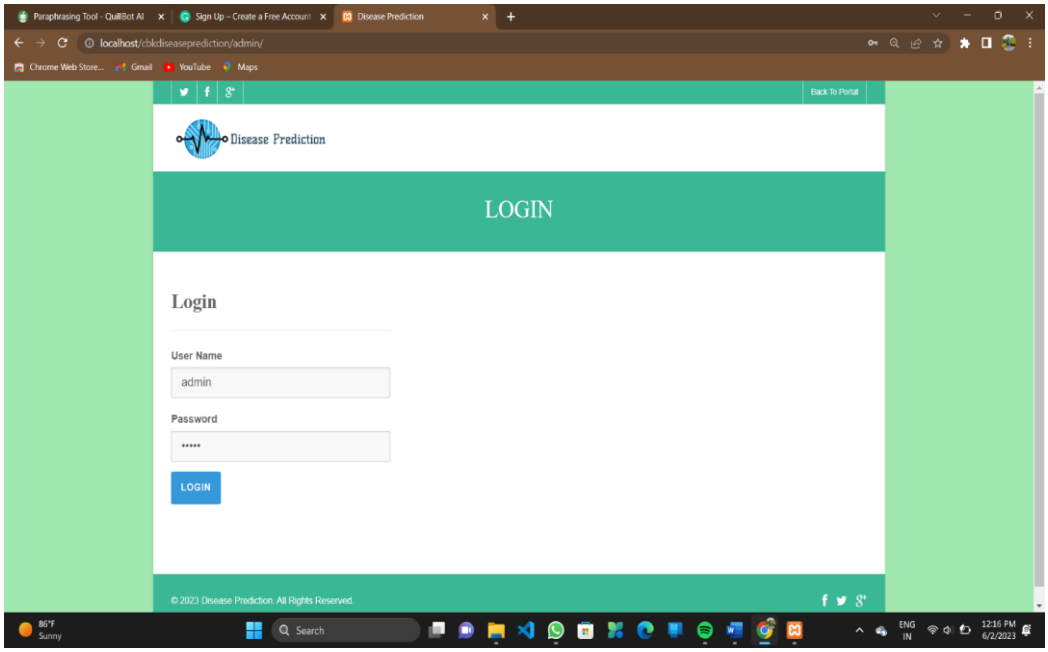

Fig.:3.1.7.3

Add Organ

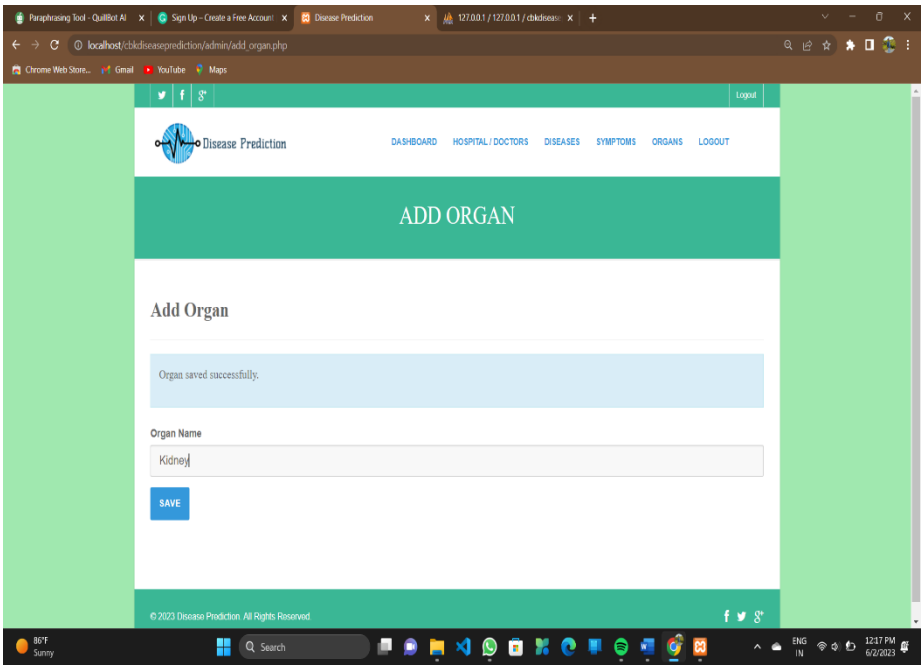

Fig.:3.1.7.4

Add Symptoms

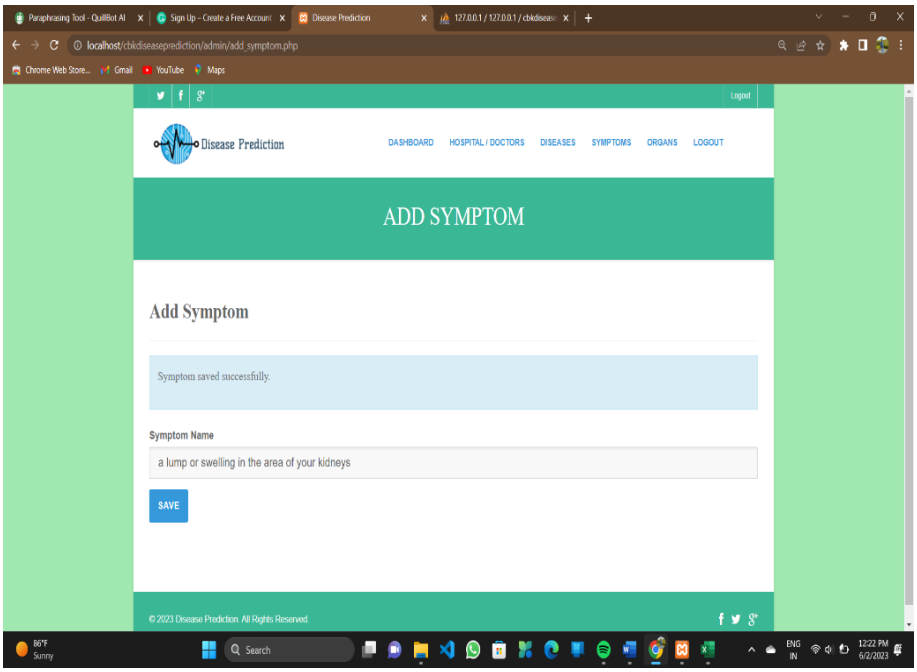

Fig.:3.1.7.5

Add Disease

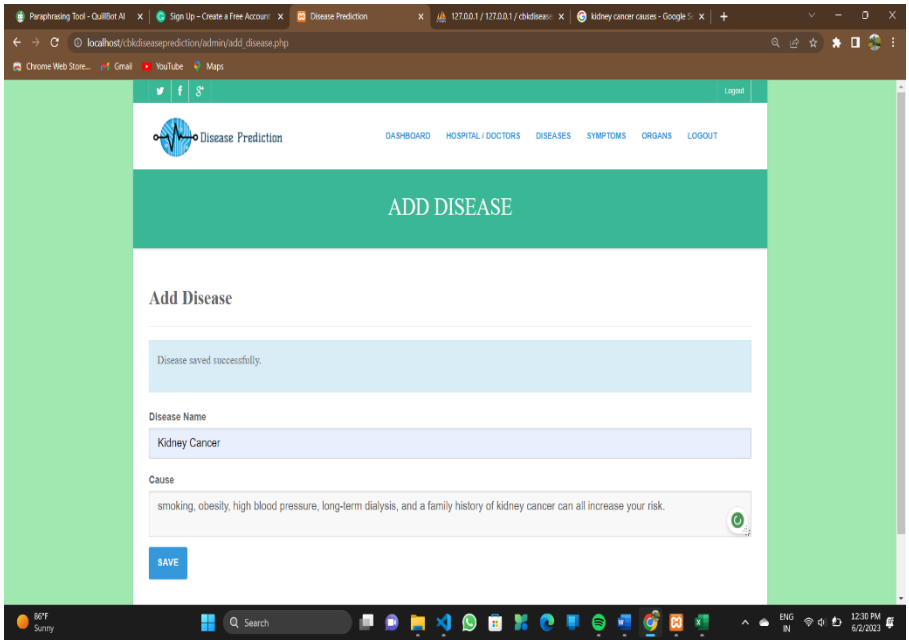

Fig.:3.1.7.6

Diseases

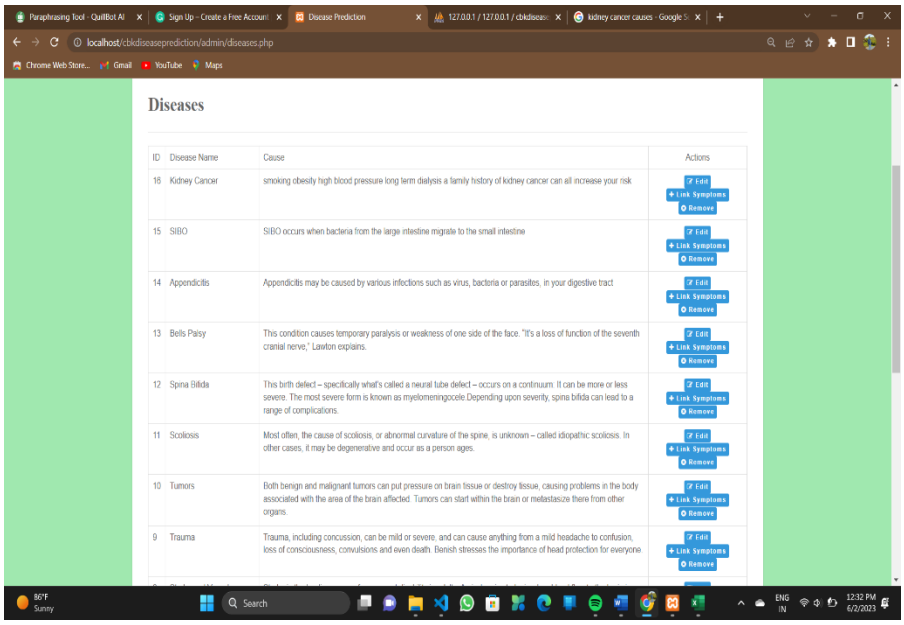

Fig.:3.1.7.7

Link Symptoms

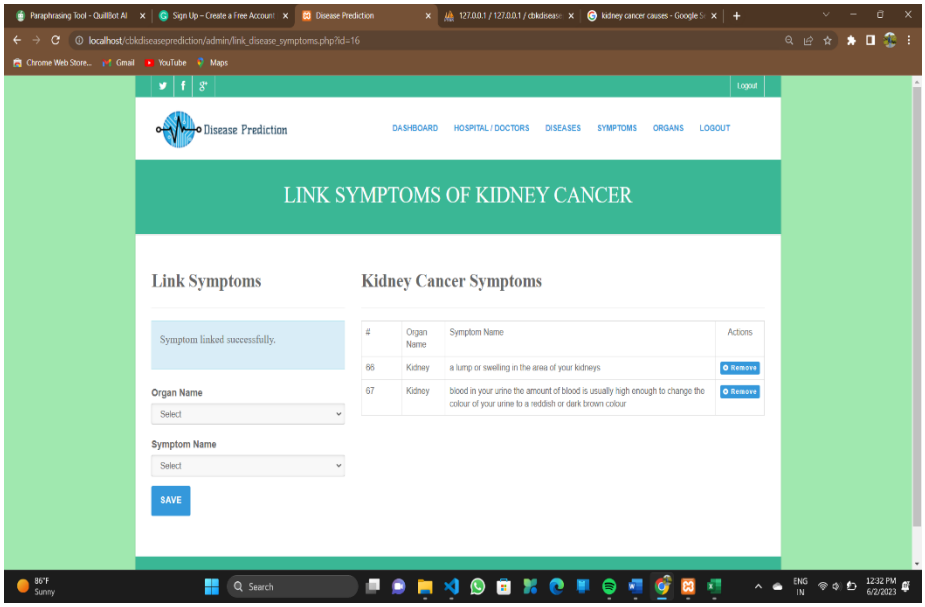

Fig.:3.1.7.8

Doctor Registration

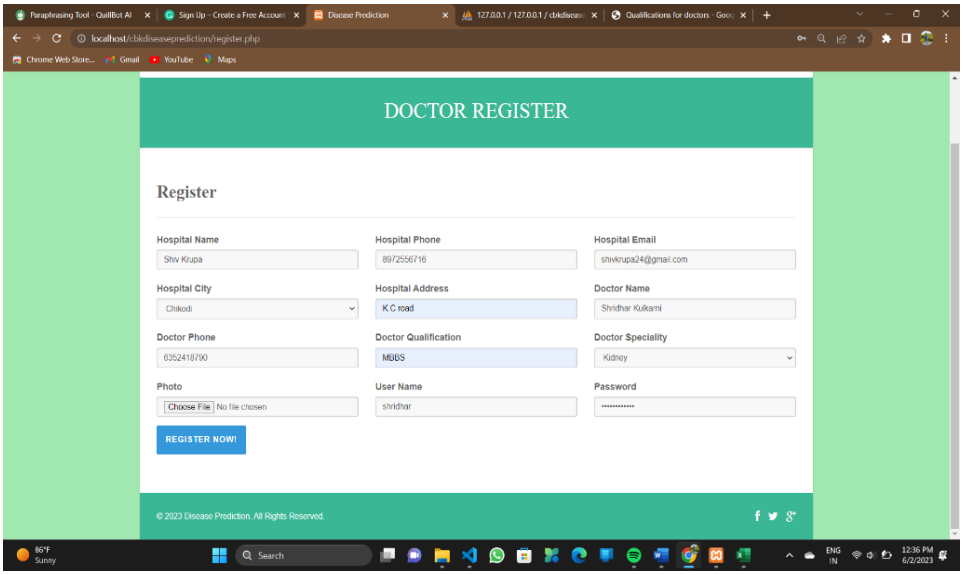

Fig.:3.1.7.9

Hospitals /Doctors

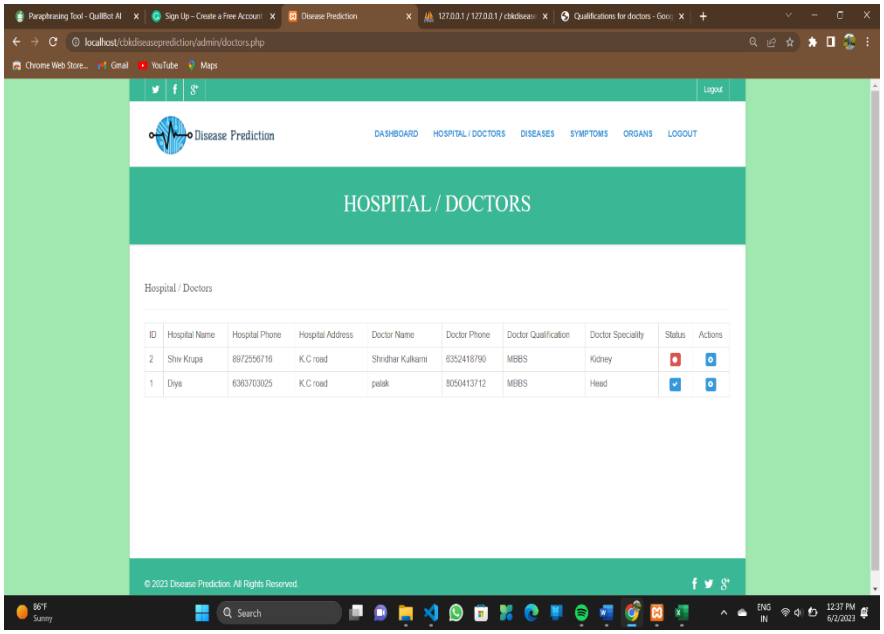

Fig.:3.1.7.10

Start Check-up

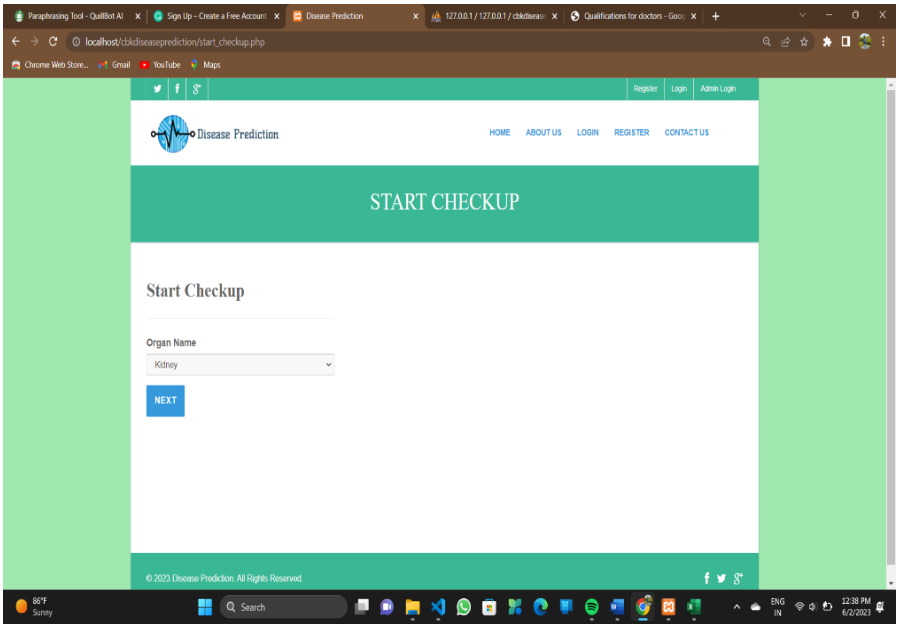

Fig.:3.1.7.11

Check Symptoms

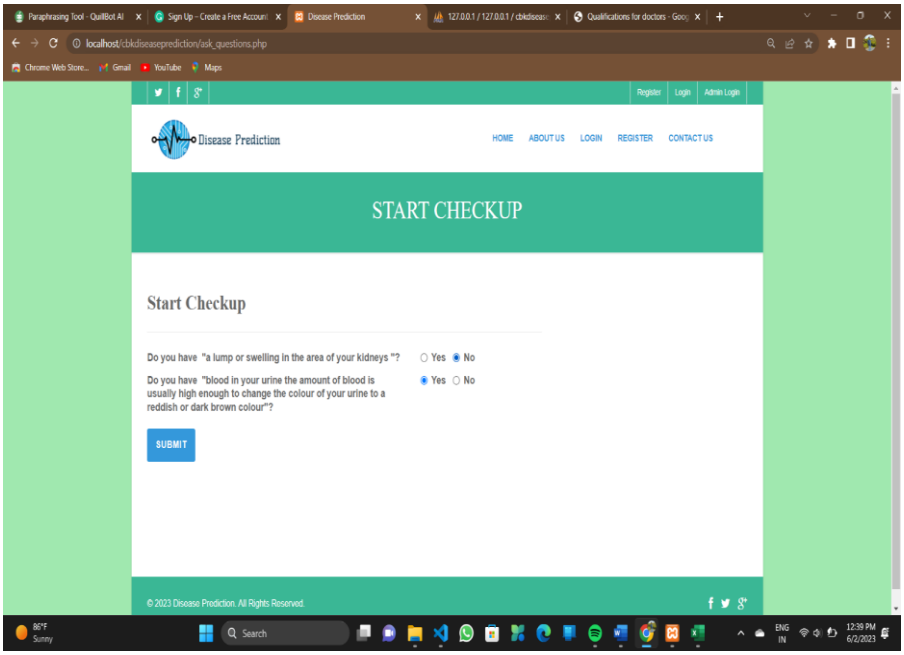

Fig.:3.1.7.12

Result

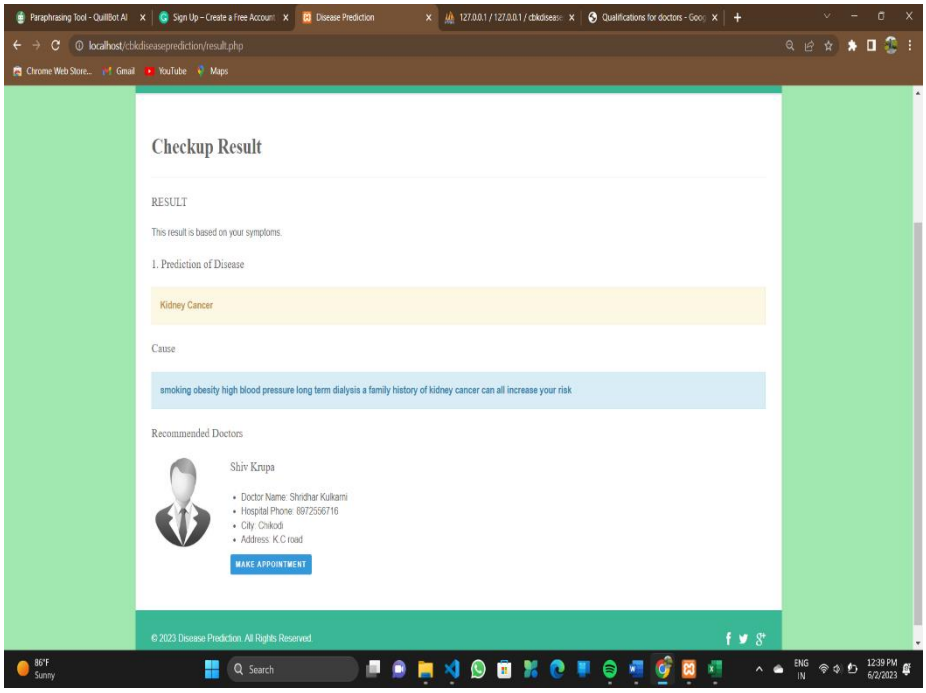

Fig.:3.1.7.13

Book Appointment

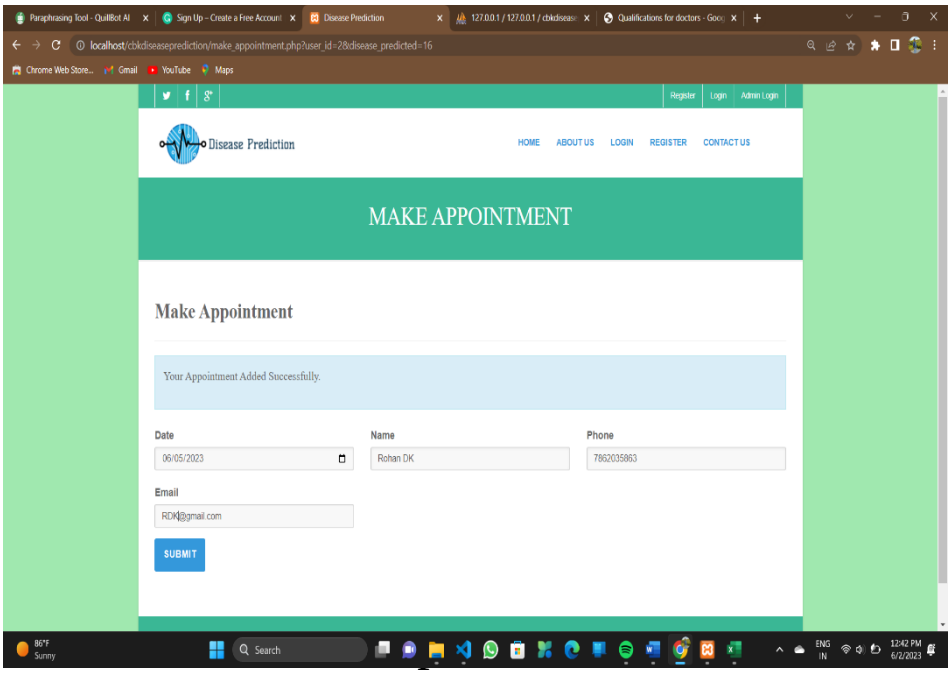

Fig.:3.1.7.14

Doctor Login

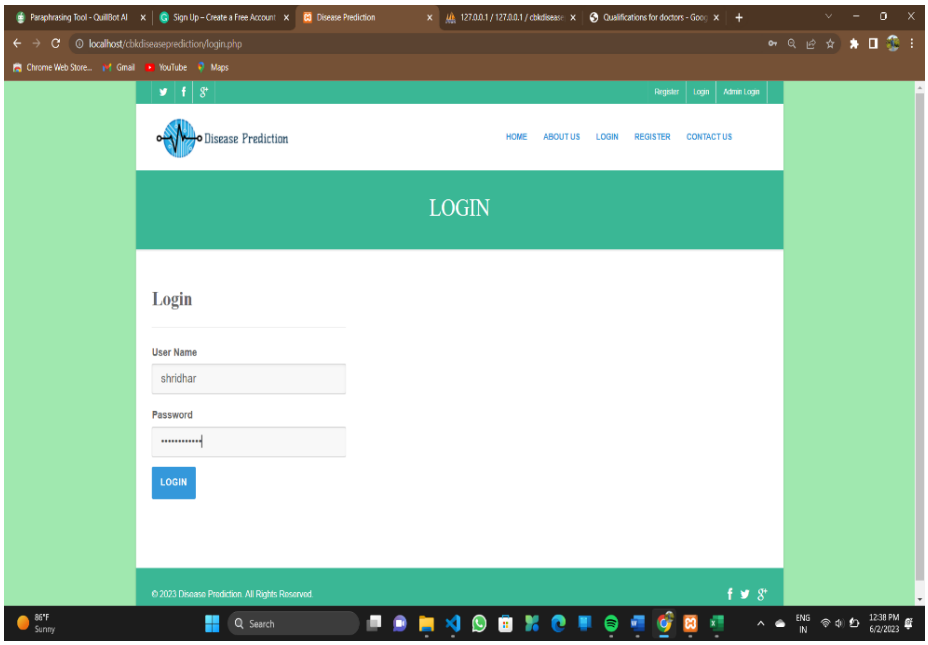

Fig.:3.1.7.15

Appointments

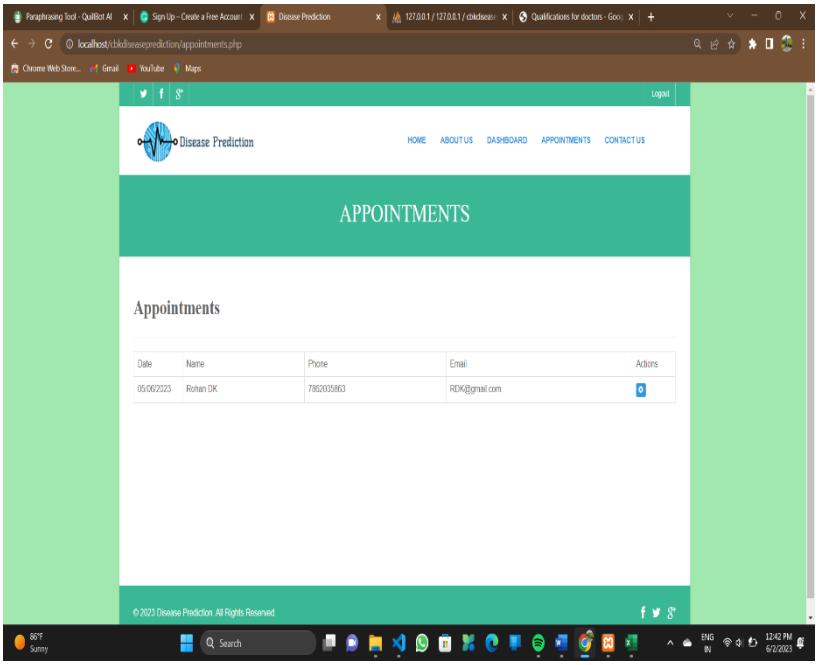

Fig.:3.1.7.16

3.1.8 Fabrication:

• Admin Login

We first gathered the prerequisites for admin login, then we analysed the information we had. The admin login form is then created. The page is validated after it has been designed, and each field of the form needs to be filled out correctly. After that, we connected the database. After finishing all the processes, the testing is finished.

• Manage Symptoms & Disease

For symptom and illness management We initially gathered requirements, then analysed those requirements. Then we built the add symptoms and add disease pages. After constructing both pages, it is validated where the symptoms are defined to fit the disease. Then we connect to a database. Finally, after completing all of the stages, testing is completed.

• Check Symptoms & Disease

We first gathered the criteria for the do Check Symptoms & disease, analysed those needs, and then designed the form. Following the evaluation of the designed page, we link to the database, and finally all the steps are finished. The testing is over.

• Predict Disease

In order to predict disease, we first collected the necessary information, which we then analysed. then the form was created. The website is validated when the disease is predicted once it has been designed. then we connect to the database. and lastly, once all the stages have been completed, testing is finished.

• Manage Doctor

For Managing Physician, we initially gathered the requirements, then we analysed them. then the form for managing the doctors was created. The page is evaluated after it has been designed. then we connect to the database. and lastly, once all the stages have been completed, testing is finished.

• Doctor Login

We first gathered the criteria for the doctor login, analyzed those needs, and then designed the form for the doctor login. Following the evaluation of the designed page, we link to the database, and finally all the steps are finished. The testing is over.

• View Doctor

For view doctor, we first gathered the requirements, then analyzed them, and finally designed the form. Following the evaluation of the designed page, we link to the database, and finally all the steps are finished. The testing is over.

• Book Appointment

For making an appointment book. collected the needs, examined them, and then created the form that has been. users have scheduled appointments with doctors. Following When all of the stages are complete and the testing is complete, the evaluation of the design page will link to the database.

• View Appointment

Gather the requirements for seeing an appointment first, analyze them, and then create the form that users will use to view the appointment. Next, doctors Check the appointment on his form by going there. Set the time as he chooses. When all testing stages are complete, the evaluation of the design page will link to the database.

System Architectural Dig.

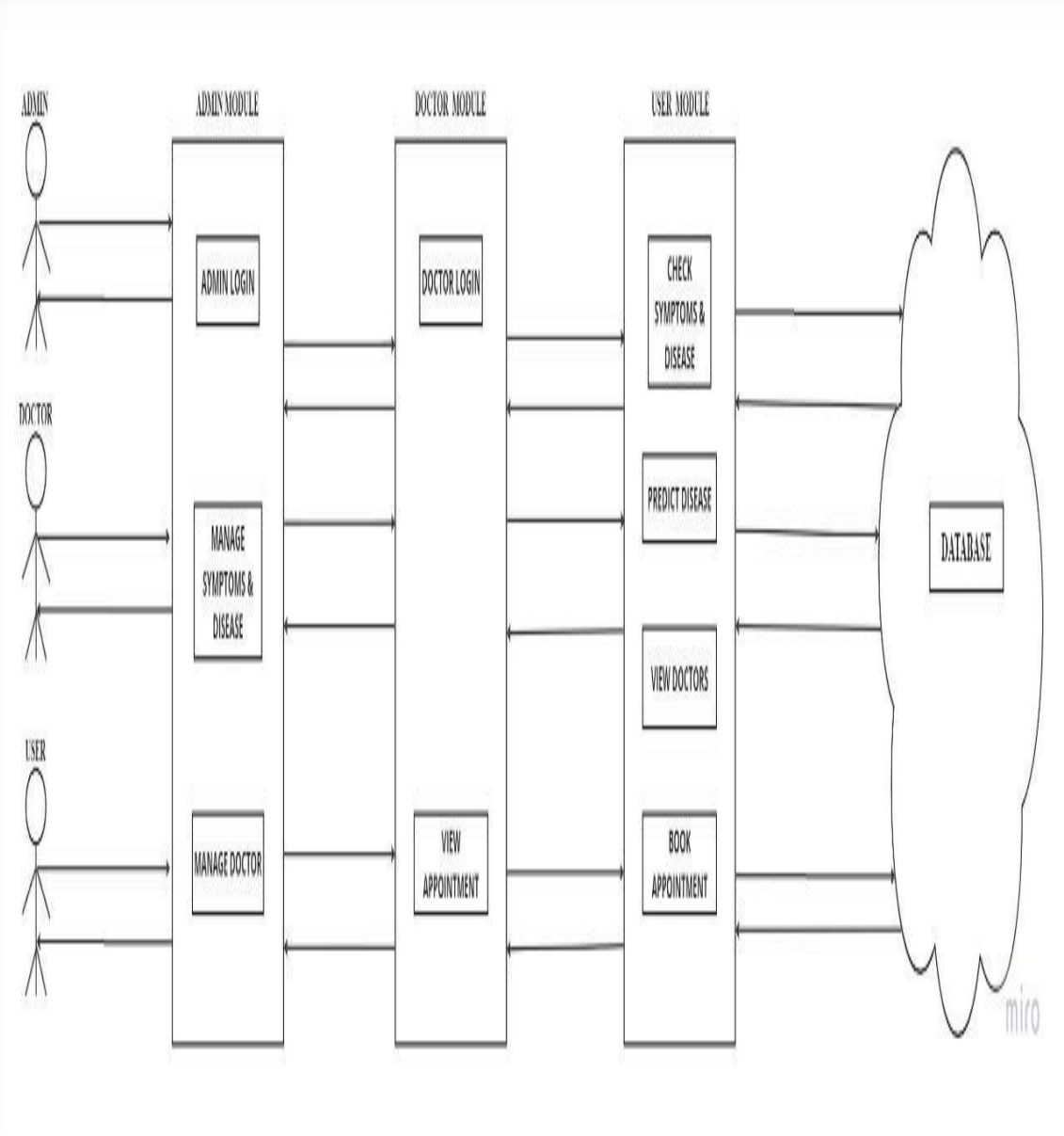

Fig.:3.1.8.1

CHAPTER 4

4.1 TEST AND VALIDATION

4.1.1 Test Plan:

Software testing is the process of confirming and validating whether a piece of software or application is bug-free, complies with all technical specifications established during its design and development, and effectively and efficiently satisfies user requirements while handling all exceptional and boundary cases.

There are two stages to software testing:

1. Verification: This is the group of procedures that make sure the programme carries out a certain function appropriately.

2. Validation is a new set of procedures that guarantee the programme is traceable to client requirements.

Testing types:

Manual Testing: -

Manual Testing is a kind of software testing in which a software tester develops and executes the test cases without using any automated testing tools. The main objective of manual testing is to detect the issues, bugs, and defects of a software application. Any new software application should be manually tested before performing the automation testing. The software testing fundamental “100% Automation is not possible” makes Manual Testing essential.

Unit Testing: -

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually scrutinized for proper operation. Software developers and sometimes QA staff complete unit tests during the development process. The main objective of unit testing is to isolate written code to test and determine if it works as intended. Unit testing is an important step in the development process. If done correctly, unit tests can detect early flaws in code which may be more difficult to find in later testing stages.

Integration Testing: -

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. Integration testing is conducted to evaluate the compliance of a system or component with specified functional requirements. It occurs after unit testing and before system testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

White Box Testing: -

White box testing is a form of application testing that provides the tester with complete knowledge of the application being tested, including access to source code and design documents. This in-depth visibility makes it possible for white box testing to identify issues that are invisible to Gray and black box testing.

Black Box Testing: -

Black box testing involves testing a system with no prior knowledge of its internal workings. A tester provides an input, and observes the output generated by the system under test. This makes it possible to identify how the system responds to expected and unexpected user actions, its response time, usability issues and reliability issues.

4.1.2 Test Approach:

Admin Login: -

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
TC01	Check Admin Login with valid data	Enter username Enter password	Username: admin Password: admin	User should login into an application	As Excepted	Pass

Table.:4.1.2.1

Manage Symptoms and Disease: -

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
TC02	While Adding the Organs, Symptoms& Disease there should not be any Numeric Values	Enter Organs Enter Symptoms Enter Disease	Organ: Kidney Symptoms: swelling in the area of your kidneys Disease: Kidney Cancer	User should enter the correct details for organs, symptoms & Diseases.	As Excepted	Pass

Table.:4.1.2.2

Check Symptoms and Disease: -

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
TC03	Selection of the organs & symptoms must be correctly done.	Selecting the organ. Selecting the Symptoms.	Organ: Kidney Symptoms: swelling in the area of your kidneys	User should correctly select the organs & Symptoms.	As Excepted	Pass

Table.:4.1.2.3

Predict Disease: -

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
TC04	While Predicting Disease the symptoms must be correctly listed out to properly Predict the disease	Selecting the Symptoms.	Symptoms: swelling in the area of your kidneys	User should correctly select the Symptoms.	As Excepted	Pass

Table.:4.1.2.4

Manage Doctor: -

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
TC05	This module is managed by the admin it will give the approval to Hospitals when they register themselves.	Username Password:	Username: Shridhar Kulkarni Password: Shridhar@1234	User should correctly Register & Login themselves	As Excepted	Pass

Table.:4.1.2.5

Doctor Login: -

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
TC06	Here the doctor will login using username & password.	Username Password	Username: Shridhar Kulkarni Password: Shridhar@1234	User should correctly Register & Login themselves	As Excepted	Pass

Table.:4.1.2.6

View Doctor: -

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
TC07	In this module the doctors will be recommended based on their symptoms Organs & Diseases.	Select Organs & Symptoms	Organ: Kidney Symptoms: swelling in the area of your kidneys	User should correctly select the organs & Symptoms	As Excepted	Pass

Table.:4.1.2.7

Book Appointment: -

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
TC08	In this module the users will be able to book the appointment based on their Diseases.	Enter appointment date & personal details like name, email & phone no.	Appointment Date: 8/05/2023 Name: Rohan. K Email-id: rohank@gmail.com Phone no. 9986472556	User should correctly fill their details	As Excepted	Pass

Table.:4.1.2.8

View Appointment: -

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
TC09	In this module the users will be able to view the appointment based on their Diseases.	Enter username Enter password	Username: Shridhar Kulkarni Password: Shridhar@1234	User should correctly put their username & password to login & view details	As Excepted	Pass

Table.:4.1.2.9

4.1.3 Features Tested:

Admin login

Manage symptoms and disease

Check symptoms and disease

Predict disease

Manage Doctor

Doctor login

View Doctor

Book appointment

View appointment

4.1.4 Features Not Tested:

Since we need to buy cloud servers, we haven't tested on live servers.

4.1.5 Finding:

- Admin Login: -

  Empty Username: If the administrator leaves the username box empty, we will get a pop-up notice that says "Please enter your username." Since we anticipated this outcome, the test is passed.

  Empty Password: If the administrator leaves the password box empty, we will get a pop-up notice that says "Please enter your password." Since we anticipated this outcome, the test is successful.

- Manage Disease & Symptoms

  Add Organs: Numerical numbers and special characters are not permitted while adding organs; if this occurs, an error message stating "Numerical values/ Special Characters can't be taken" will appear.

  Add Symptoms: Numeric numbers are permitted when adding symptoms, however the system will not forward us to the next page of the website if this occurs.

  Add Disease: It is impossible to leave the box empty while adding the disease; otherwise, the notice "The disease must be added" will appear. Only then will it successfully advance us to the next page.

- Check Symptoms & Disease

  Select Organ: The organ must be selected; else, a pop-up notification stating "The organ must be selected" will show. The organ selection process must be correctly completed.

Select Symptoms: If the patient or user doesn't have the given symptom, they can check it as no, but all the boxes must be ticked. The selection of symptoms must be done properly, and no boxes may be left unchecked.

- Predict Disease: -

  Select Organ: The organ must be selected; else, a pop-up notification stating "The organ must be selected" will show. The organ selection process must be correctly completed.

  Select Symptoms: If the patient or user doesn't have the given symptom, they can check it as no, but all the boxes must be ticked. The selection of symptoms must be done properly, and no boxes may be left unchecked.

- Manage Doctor: -

  Empty Username: If the administrator leaves the username box empty, we will get a pop-up notice that says "Please enter your username." Since we anticipated this outcome, the test is passed.

  Empty Password: If the administrator leaves the password box empty, we will get a pop-up notice that says "Please enter your password." Since we anticipated this outcome, the test is successful.

- Doctor Login: -

  Incomplete Username: If the doctor leaves the username box blank, we will get a pop-up message that says "Please enter your username." The test succeeded because we anticipated this outcome.

  Empty Password: If the Doctor leaves the password box blank, we will receive a pop-up message stating, "Please enter your password." The test was successful because we expected this conclusion.

- View Doctor: -

  Choose Organ: The organ must be selected; else, a pop-up message stating "The organ must be selected" will appear. The organ selection procedure must be followed appropriately.

  Choose Symptoms: If the patient or user does not have the specified symptom, they can mark it as no, but all boxes must be checked. The symptoms must be appropriately selected, and no boxes may be left unchecked.

- Book Appointment: -

Appointment Date: When entering the date on the appointment page, Alphabetical characters are not accepted; if this occurs, an error message stating "Alphabets are not allowed" will show.

Name: When entering a name, special characters and numbers are not permitted; if this occurs, the user will be unable to proceed or the appointment will be cancelled.

Email-id: The email-id must be filled out and in the proper format; otherwise, the user will be unable to progress or the appointment will be cancelled.
Phone number: no alphabets or special characters are permitted; if this occurs, an error message will be shown and we will not be able to go to the next page.

- View Appointment: -

  Incomplete Username: If the doctor leaves the username box unfilled, we will receive a pop-up message saying "Please enter your username." The test was successful because we expected it to be successful.

  Blank Password: If the Doctor leaves the password field empty, we will get a pop-up notification that says, "Please enter your password." We predicted this result, hence the test was successful.

4.1.6 Inference:

- Admin Login: The admin module is the most crucial one in this case because it controls all dependent modules. The other two main modules, the Doctor module and the User module, are all handled by this module.
- Manage Symptoms & Disease: The Manage Symptoms & Disease module depends on the admin logging in because doing so adds organs, symptoms, and diseases as well as links all of them to their respective diseases.
- Check Symptoms & Diseases: Checking symptoms and diseases is dependent on managing symptoms and diseases since users can only check diseases when symptoms and diseases are handled by the admin.
- Predict Disease: Predicting disease is dependent on checking symptoms and diseases because when symptoms and organs are recorded, doctors are recommended.

- Manage Doctor: This module is managed by admin and is depending on disease prediction. When an illness is predicted, the administrator oversees the doctors, and the system recommends doctors depending on disease.
- Doctor Login: Doctor login is handled by admin login, and this module is dependent on manage doctors; when physicians register themselves, the admin accepts hospital/doctor registration.
- View Doctor: View Doctor is entirely dependent on the doctor login because only the doctors/hospitals who login and specify their details will be recommended based on that disease.
- Book Appointment: Booking an appointment is based on doctor recommendations because when doctors are suggested, the patient or user can book an appointment.
- View Appointment: To view the appointments, people must first book them, and then doctors must login to their portal to view them.

CHAPTER-5

5.1 BUSINESS ASPECTS

5.1.1 The market and economic outlook of the caption project

In order to implement this jack, we're going to use full stack development languages like PHP, JavaScript, Bootstrap, MySQL DB, HTML, and JavaScript. Doctors will need to register to this portal, and users can visit it. Based on user symptoms, the system predicts the user's disease and at that time, the system will recommend the available doctors through this portal. The user will get the best doctor for his disease.

5.1.2 Highlight the novel features of Project/Service

User will get disease information any time.

Reduce the time of user.

User will get Doctor recommendations easily.

User will get disease information anywhere.

5.1.3 How does a product/service fit into a competitive landscape

The healthcare industry will benefit from this initiative because there is now no website or application that can be of this kind. is a medical application that aids in illness prediction and suggests a qualified physician for the patient.

It successfully predicts illnesses using big data analytics and machine learning techniques.

The application also offers details on the doctor, including their qualifications, experience, and patient testimonials.

The user may use this to decide which doctor they should contact in an educated manner.

Since there isn't a similar website or application and because it is really useful and may save a lot of time, the government and healthcare sectors will be among our clients.

5.1.4 Who are the possible capital project clients/customer

we are going to develop this is prediction and doctor recommendation the purpose of this project is create Disease prediction online and recommendation in this project doctor needs to register to this portal and user can visit to this portal and as per user symptoms system predicts the user disease And that time system will recommend the available doctor through this portal User will get perfect doctor for his disease to implement this jack we're going to use full stack development languages like PHP, JavaScript, Bootstrap, MySQL DB, HTML and CSS.

5.2 FINANCIAL CONSIDERATIONS

5.2.1 Capstone project budget:

The total estimated cost of this project ₹48,609/-

5.2.2 Cost capstone projections needed for either for profit/non-profit options.

Estimated Cost:	₹48,609/-
Labour Cost:	₹27,510/-
Material Cost:	₹8,444/-
Profit of the Project: ₹12655/-	

Table.:5.2.2.1

5.3 CONCLUSION AND RECOMMENDATIONS

5.3.1 Describe state of complement of caption project

Our skilled and experienced team will design and create a website that fully adheres to all pre-established system standards. The website will boast a range of expertly crafted components, including a powerful disease prediction module, robust user and patient management tools, highly effective doctor management features, and intuitive doctor recommendation systems. By leveraging the advanced capabilities of the disease prediction module, visitors to the site will be able to accurately assess their overall health status, while at the same time receiving personalized recommendations on appropriate disease management strategies and being connected with best-fit specialist doctors for their individual needs.

5.3.2 Future Work: -

We have the flexibility to allocate the project in any city or state, depending on the available resources and logistical feasibility.

In the future, it is possible for us to expand the current idea and transform it into a mobile application.

Program Code

- **Index. Php**

```
<!DOCTYPE HTML>
<html class="no-js">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title>Disease Prediction</title>
<meta name="viewport" content="width=device-width, user-scalable=no,
initial-scale=1.0, minimum-scale=1.0, maximum-scale=1.0"/>
<meta name="format-detection" content="telephone=no"/>
<link href="css/bootstrap.css" rel="stylesheet" type="text/css"/>
<link href="css/style.css" rel="stylesheet" type="text/css"/>
<link href="plugins/sequence/css/sequence.html" rel="stylesheet"
type="text/css"/>
<link href="plugins/prettyphoto/css/prettyPhoto.css" rel="stylesheet"
type="text/css"/>
<link href="plugins/owl-carousel/css/owl.carousel.css" rel="stylesheet"
type="text/css"/>
<link href="plugins/owl-carousel/css/owl.theme.css" rel="stylesheet"
type="text/css"/>
<link rel="stylesheet" type="text/css" href="plugins/rs-
plugin/css/settings.css" media="screen" />
<!--[if lte IE 8]><link rel="stylesheet" type="text/css" href="css/ie8.css"
media="screen" /><![endif]-->
<link class="alt" href="colors/blue.css" rel="stylesheet" type="text/css"/>
<script src="js/modernizr.js"></script>
<?php
require_once "healthhelper.php";
$helper = new HealthHelper();
?>
</head>
```

```
<body class="boxed"  style="background-image: url('images/backgrounds/images/img3.jpg'); background-repeat: no-repeat; background-size: cover;">
<!--[if lt IE 7]>
    <p class="chromeframe">You are using an outdated browser. <a href="http://browsehappy.com/">Upgrade your browser today</a> or <a href="http://www.google.com/chromeframe/?redirect=true">install Google Chrome Frame</a> to better experience this site.</p>
<![endif]-->
<!-- Start Body Container -->
<div class="body footer-style2">
 <!-- Start Header -->
 <?php
  require_once "header.php";
 ?>
 <!-- End Header -->
 <!-- Start Content -->
 <div class="main" role="main">
  <div id="content" class="content full">
   <div class="rev-slider-container">
    <div class="tp-banner-container">
     <div class="tp-banner" >
      <ul>
       <!-- SLIDE  -->
       <li data-delay="4000" data-masterspeed="600" data-slotamount="7" data-transition="scaledownfromtop">
        <!-- MAIN IMAGE -->
        <img src="images/slide1.jpg" alt="">
       </li>

       <li data-delay="4000" data-masterspeed="600" data-slotamount="7" data-transition="scaledownfromtop">
        <!-- MAIN IMAGE -->
        <img src="images/slide2.jpg" alt="">
```

```
            </li>
            </ul>
          </div>
        </div>
      </div>
      <div class="featured-row dgray-color margin-20">
        <div class="container">
          <div class="row">
            <div class="col-md-12">
              <h2>Welcome to Disease Prediction.</h2>
              <a href="start_checkup.php" class="btn btn-primary">Start
Checkup</a> </div>
          </div>
        </div>
      </div>
      <div class="container margin-30">
        <div class="row">
          <div class="col-md-12 text-align-center">
            <h2>About Us</h2>
            <h4 class="heading-hr no-strong accent-color-text margin-
50"><span>Few words That Helps To Understand</span></h4>
          </div>
          <div class="col-md-4 col-sm-4">
            <p class="drop-caps">The aim of this study is to develop a disease
prediction and doctor recommendation system using machine learning
techniques. The proposed system will analyze the patient's medical
history, symptoms, and other relevant information to predict the likelihood
of various diseases. Based on the predicted disease, the system will
recommend a list of doctors specialized in the relevant field. The system
will use a large dataset of patient medical records to train and validate the
machine learning model. The proposed system has the potential to
improve healthcare outcomes by providing timely and accurate disease
predictions, as well as recommending the most appropriate doctors for
patients.</p>
```

```
</div>
<div class="col-md-4 col-sm-4">
  <p>Disease prediction and doctor recommendation systems are a rapidly growing area of healthcare technology, leveraging the power of machine learning to improve the accuracy and efficiency of medical diagnosis and treatment. By analyzing patient data and symptoms, these systems can predict the likelihood of various diseases and recommend specialized doctors for treatment. Such systems have the potential to revolutionize healthcare delivery by improving patient outcomes, reducing healthcare costs, and enhancing the overall patient experience.</p>
</div>
<div class="col-md-4 col-sm-4">
  <!-- Start Accordion -->
  <div class="accordion" id="accordionArea">
    <div class="accordion-group panel">
      <div class="accordion-heading accordionize"> <a class="accordion-toggle" data-toggle="collapse" data-parent="#accordionArea" href="#oneArea"> Our Vision <i class="fa fa-angle-down"></i> </a> </div>
        <div id="oneArea" class="accordion-body collapse">
          <div class="accordion-inner">
The vision for the disease prediction and doctor recommendation system is to create a healthcare ecosystem that is predictive, proactive, and patient-centric. By leveraging the latest advances in machine learning and artificial intelligence, this system aims to enable doctors to make more accurate diagnoses, reduce misdiagnosis rates, and improve patient outcomes. Furthermore, the system aims to empower patients by providing them with personalized disease predictions and recommendations for specialized doctors, thus enabling them to take control of their health and make informed decisions about their treatment. Ultimately, the goal is to transform the healthcare industry by delivering more efficient, effective, and patient-centered care.</div>
        </div>
      </div>
```

  Our Philosophy

 The philosophy behind the disease prediction and doctor recommendation system is based on the principles of personalized medicine and patient-centered care. The system is designed to leverage the power of machine learning to provide tailored disease predictions and doctor recommendations based on individual patient data, symptoms, and medical history. The system aims to facilitate more accurate diagnoses, reduce misdiagnosis rates, and provide patients with access to the most suitable doctors for their condition. Furthermore, the system emphasizes the importance of patient autonomy, by empowering patients to take control of their health and make informed decisions about their treatment. The ultimate goal is to improve the quality of healthcare delivery by providing more accurate, personalized, and patient-centered care.

  Our Mission

 The vision for the disease prediction and doctor recommendation system is to create a healthcare ecosystem that is predictive, proactive, and patient-centric. By leveraging the latest advances in machine learning and artificial intelligence, this system aims to enable doctors to make more accurate diagnoses, reduce misdiagnosis rates, and improve patient outcomes. Furthermore, the system aims to empower patients by providing them with personalized

disease predictions and recommendations for specialized doctors, thus enabling them to take control of their health and make informed decisions about their treatment. Ultimately, the goal is to transform the healthcare industry by delivering more efficient, effective, and patient-centered care.

```
            </div>
          </div>
        </div>
        <!-- End Accordion -->
      </div>
    </div>
    </div>
 </div>
 <?php
 require_once "footer.php";
 ?>
 </div>
<!-- End Body Container -->
<script src="js/jquery-latest.min.js"></script> <!-- Jquery Library Call -->
<script src="plugins/prettyphoto/js/prettyphoto.js"></script>
<script src="plugins/owl-carousel/js/owl.carousel.min.js"></script>
<script src="plugins/page-scroller/jquery.pagescroller.js"></script>
<script src="js/helper-plugins.js"></script> <!-- Plugins -->
<script src="js/bootstrap.js"></script> <!-- UI -->
<script src="js/init.js"></script> <!-- All Scripts -->
<script src="plugins/rs-
plugin/js/jquery.themepunch.plugins.min.js"></script>
<script src="plugins/rs-
plugin/js/jquery.themepunch.revolution.min.js"></script>
<script src="js/revolution-slider-init.js"></script> <!-- Revolutions Slider
Intialization -->
<!-- Preloader -->
<script type="text/javascript">
```

```
//<![CDATA[
$(window).load(function() { // makes sure the whole site is loaded
$("#status").fadeOut(); // will first fade out the loading animation
$("#preloader").delay(350).fadeOut("slow"); // will fade out the white
DIV that covers the website.});
//]]>
</script>
<!-- End Js -->
</body>
</html>
```

- Admin Login

```
<!DOCTYPE HTML>
<html class="no-js">
<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title>Disease Prediction</title>

<meta name="viewport" content="width=device-width, user-scalable=no,
initial-scale=1.0, minimum-scale=1.0, maximum-scale=1.0"/>
<meta name="format-detection" content="telephone=no"/>

<link href="../css/bootstrap.css" rel="stylesheet" type="text/css"/>
<link href="../css/style.css" rel="stylesheet" type="text/css"/>
<link href="../plugins/sequence/css/sequence.html" rel="stylesheet"
type="text/css"/>
<link href="../plugins/prettyphoto/css/prettyPhoto.css" rel="stylesheet"
type="text/css"/>
<link href="../plugins/owl-carousel/css/owl.carousel.css" rel="stylesheet"
type="text/css"/>
<link href="../plugins/owl-carousel/css/owl.theme.css" rel="stylesheet"
type="text/css"/>
```

```
<link rel="stylesheet" type="text/css" href="../plugins/rs-
plugin/css/settings.css" media="screen" />

<!--[if lte IE 8]><link rel="stylesheet" type="text/css" href="../css/ie8.css"
media="screen" /><![endif]-->


<link class="alt" href="../colors/blue.css" rel="stylesheet"
type="text/css"/>

<script src="../js/modernizr.js"></script>

<script type="text/javascript">
   function validate_form()
   {
      var username = document.getElementById("username").value;
      var password = document.getElementById("password").value;

      if(username=='')
      {
         alert("Please Enter User Name.");
         return false;
      }
      else if(password=='')
      {
         alert("Please Enter Password.");
         return false;
      }
   }
</script>
<?php
require_once "adminhelper.php";
$helper = new AdminHelper();
if($_SESSION['admin_userid']!='')
```

```
{
   echo "<script>window.location='dashboard.php';</script>";
}
?>

</head>
<body class="boxed"  style="background-image: url('../images/backgrounds/images/img3.jpg'); background-repeat: no-repeat; background-size: cover;">
<!--[if lt IE 7]>
     <p class="chromeframe">You are using an outdated browser. <a href="http://browsehappy.com/">Upgrade your browser today</a> or <a href="http://www.google.com/chromeframe/?redirect=true">install Google Chrome Frame</a> to better experience this site.</p>
<![endif]-->
<!-- Preloader -->
<div id="preloader">
  <div id="status"></div>
</div>

<!-- Start Body Container -->
<div class="body footer-style2">
  <!-- Start Header -->
  <?php
    require_once "header.php";
  ?>
  <!-- End Header -->

  <!-- Start Content -->
  <div class="main" role="main">
    <div id="content" class="content page-content full">
      <header class="page-header flexible parallax text-align-center parallax-overlay" style="background-image:url(../images/img4.jpg)">
        <section>
```

```
        <div class="container">
          <div class="row">
            <div class="col-md-12">
              <h1>Admin Login</h1>
            </div>
          </div>
        </div>
      </section>
    </header>
    <div class="container">
      <div class="row">
        <div class="col-md-4">
          <h2><strong>Admin Login</strong></h2>
          <hr/>
          <?php
          if($_GET['error'])
          {
             ?>
             <div class="alert alert-info" role="alert">
                                        <h3 style="text-transform: none;">Login
Failed! Please enter valid details.</h3>
                                    </div>
             <?php
          }
          ?>
          <form method="post" action="checkuser.php" onsubmit="return
validate_form();">
            <div class="row">
              <div class="form-group">
                <div class="col-md-12">
                  <label>User Name</label>
                  <input type="text" id="username" name="username"
class="form-control input-lg" placeholder="Username">
                </div>
```

```
            </div>
          </div>
          <div class="row">
            <div class="form-group">
              <div class="col-md-12">
                <label>Password</label>
                <input type="password" id="password" name="password"
class="form-control input-lg" placeholder="Password">
              </div>
            </div>
          </div>
          <div class="row">
            <div class="form-group">
              <div class="col-md-12">
                <input type="submit" name="submit" class="btn btn-primary
btn-lg" value="Login">
              </div>
            </div>
          </div>
        </form>
      </div>
      <!-- Start Sidebar -->
      <aside class="col-md-3 sidebar right-sidebar">


      </aside>
    </div>
  </div>
</div>

<?php
 require_once "footer.php";
?>
```

```
</div>
<!-- End Body Container -->
<script src="../js/jquery-latest.min.js"></script> <!-- Jquery Library Call -
->
<script src="../plugins/prettyphoto/js/prettyphoto.js"></script>
<script src="../plugins/prettyphoto/js/prettyphoto.js"></script>
<script src="../plugins/owl-carousel/js/owl.carousel.min.js"></script>
<script src="../plugins/page-scroller/jquery.pagescroller.js"></script>
<script src="../js/helper-plugins.js"></script> <!-- Plugins -->
<script src="../js/bootstrap.js"></script> <!-- UI -->
<script src="../js/init.js"></script> <!-- All Scripts -->
<!-- Preloader -->
<script type="text/javascript">
    //<![CDATA[
            $(window).load(function() { // makes sure the whole site is
loaded
                $("#status").fadeOut(); // will first fade out the loading
animation
                $("#preloader").delay(350).fadeOut("slow"); // will fade
out the white DIV that covers the website.
            });
    //]]>
</script>
<!-- End Js -->
</body>
</html>
```

- Doctor Login

```
<!DOCTYPE HTML>
<html class="no-js">
<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title>Disease Prediction</title>
```

```
<meta name="viewport" content="width=device-width, user-scalable=no,
initial-scale=1.0, minimum-scale=1.0, maximum-scale=1.0"/>
<meta name="format-detection" content="telephone=no"/>

<link href="css/bootstrap.css" rel="stylesheet" type="text/css"/>
<link href="css/style.css" rel="stylesheet" type="text/css"/>
<link href="plugins/sequence/css/sequence.html" rel="stylesheet"
type="text/css"/>
<link href="plugins/prettyphoto/css/prettyPhoto.css" rel="stylesheet"
type="text/css"/>
<link href="plugins/owl-carousel/css/owl.carousel.css" rel="stylesheet"
type="text/css"/>
<link href="plugins/owl-carousel/css/owl.theme.css" rel="stylesheet"
type="text/css"/>
<link rel="stylesheet" type="text/css" href="plugins/rs-
plugin/css/settings.css" media="screen" />

<!--[if lte IE 8]><link rel="stylesheet" type="text/css" href="css/ie8.css"
media="screen" /><![endif]-->


<link class="alt" href="colors/blue.css" rel="stylesheet" type="text/css"/>

<script src="js/modernizr.js"></script>

<script type="text/javascript">
   function validate_form()
   {
      var username = document.getElementById("username").value;
      var password = document.getElementById("password").value;

      if(username=="")
      {
```

```
            alert("Please Enter User Name.");
            return false;
        }
        else if(password=='')
        {
            alert("Please Enter Password.");
            return false;
        }
    }
</script>
<?php
require_once "healthhelper.php";
$helper = new HealthHelper();
?>

</head>
<body class="boxed"  style="background-image:
url('images/backgrounds/images/img3.jpg'); background-repeat: no-
repeat; background-size: cover;">
<!--[if lt IE 7]>
    <p class="chromeframe">You are using an outdated browser. <a
href="http://browsehappy.com/">Upgrade your browser today</a> or <a
href="http://www.google.com/chromeframe/?redirect=true">install
Google Chrome Frame</a> to better experience this site.</p>
<![endif]-->
<!-- Preloader -->
<div id="preloader">
  <div id="status"></div>
</div>

<!-- Start Body Container -->
<div class="body footer-style2">
  <!-- Start Header -->
  <?php
```

```
  require_once "header.php";
 ?>
 <!-- End Header -->

 <!-- Start Content -->
 <div class="main" role="main">
  <div id="content" class="content page-content full">
   <header class="page-header flexible parallax text-align-center
parallax-overlay" style="background-image:url(images/img4.jpg)">
     <section>
      <div class="container">
       <div class="row">
        <div class="col-md-12">
         <h1>Doctor Login</h1>
        </div>
       </div>
      </div>
     </section>
    </header>
    <div class="container">
     <div class="row">
      <div class="col-md-4">
       <h2><strong>Doctor Login</strong></h2>
       <hr/>
       <?php
       if($_GET['error'])
       {
          ?>
          <div class="alert alert-info" role="alert">
                                    <h3 style="text-transform: none;">Login
Failed! Please enter valid details.</h3>
                                </div>
          <?php
       }
```

```
        ?>
        <form method="post" action="checkuser.php" onsubmit="return
validate_form();">
          <div class="row">
            <div class="form-group">
              <div class="col-md-12">
                <label>User Name</label>
                <input type="text" id="username" name="username"
class="form-control input-lg" placeholder="Username">
              </div>
            </div>
          </div>
          <div class="row">
            <div class="form-group">
              <div class="col-md-12">
                <label>Password</label>
                <input type="password" id="password" name="password"
class="form-control input-lg" placeholder="Password">
              </div>
            </div>
          </div>
          <div class="row">
            <div class="form-group">
              <div class="col-md-12">
                <input type="submit" name="submit" class="btn btn-primary
btn-lg" value="Login">
              </div>
            </div>
          </div>
        </form>
      </div>
      <!-- Start Sidebar -->
      <aside class="col-md-3 sidebar right-sidebar">
```

```
        </aside>
       </div>
     </div>
   </div>
  </div>

  <?php
   require_once "footer.php";
  ?>
</div>
<!-- End Body Container -->
<script src="js/jquery-latest.min.js"></script> <!-- Jquery Library Call -->
<script src="plugins/prettyphoto/js/prettyphoto.js"></script>
<script src="plugins/prettyphoto/js/prettyphoto.js"></script>
<script src="plugins/owl-carousel/js/owl.carousel.min.js"></script>
<script src="plugins/page-scroller/jquery.pagescroller.js"></script>
<script src="js/helper-plugins.js"></script> <!-- Plugins -->
<script src="js/bootstrap.js"></script> <!-- UI -->
<script src="js/init.js"></script> <!-- All Scripts -->
<!-- Preloader -->
<script type="text/javascript">
    //<![CDATA[
            $(window).load(function() { // makes sure the whole site is
loaded
                    $("#status").fadeOut(); // will first fade out the loading
animation
                    $("#preloader").delay(350).fadeOut("slow"); // will fade
out the white DIV that covers the website.
            });
    //]]>
</script>
<!-- End Js -->
</body>
```

```
</html>
```

- Make Appointment

```
<!DOCTYPE HTML>
<html class="no-js">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title>Disease Prediction</title>
<meta name="viewport" content="width=device-width, user-scalable=no,
initial-scale=1.0, minimum-scale=1.0, maximum-scale=1.0"/>
<meta name="format-detection" content="telephone=no"/>
<link href="css/bootstrap.css" rel="stylesheet" type="text/css"/>
<link href="css/style.css" rel="stylesheet" type="text/css"/>
<link href="plugins/sequence/css/sequence.html" rel="stylesheet"
type="text/css"/>
<link href="plugins/prettyphoto/css/prettyPhoto.css" rel="stylesheet"
type="text/css"/>
<link href="plugins/owl-carousel/css/owl.carousel.css" rel="stylesheet"
type="text/css"/>
<link href="plugins/owl-carousel/css/owl.theme.css" rel="stylesheet"
type="text/css"/>
<link rel="stylesheet" type="text/css" href="plugins/rs-
plugin/css/settings.css" media="screen" />
<!--[if lte IE 8]><link rel="stylesheet" type="text/css" href="css/ie8.css"
media="screen" /><![endif]-->
<link class="alt" href="colors/blue.css" rel="stylesheet" type="text/css"/>
<script src="js/modernizr.js"></script>
<script type="text/javascript">
    function validate_form()
    {
        var date    = document.getElementById("date").value;
        var name    = document.getElementById("name").value;
        var phone   = document.getElementById("phone").value;
        var email   = document.getElementById("email").value;
```

```
var validchar = /^[A-Z a-z]+$/;

if(date==")
{
   alert("Please Enter Date.");
   return false;
}
if (name.trim() === ") {
  alert("Please enter a name.");
  return false;
} else if (/\d/.test(name)) {
  alert("Name should not contain any numeric digits.");
  return false;
} else if (!/^[A-Za-z\s]+$/.test(name)) {
  alert("Name should only contain alphabets and spaces.");
  return false;
} else if (/([A-Za-z])\1{2,}/.test(name)) {
  alert("Name should not have a repeating single character more than 2
times.");
  return false;
} else if (/[\[\]"':-]/.test(name)) {
  alert("Name should not contain any special characters.");
  return false;
}
else if(phone==")
{
   alert("Please Enter Phone Number.");
   return false;
}
else if(isNaN(phone))
{
   alert("Phone Number should be numeric.");
   return false;
```

```
        }
        else if(checkInternationalPhone(phone)==false)
        {
            alert("Please Enter a Valid Phone Number");
                return false;
        }
        var emailRegex = /^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}$/;

        if (email.trim() === '') {
          alert("Please enter an email address.");
          return false;
        } else if (!emailRegex.test(email)) {
          alert("Please enter a valid email address.");
          return false;
        } else if (/(.)\1+@/.test(email) || /@(.+)\1+/.test(email)) {
          alert("Email address should not have repeating characters before or after the @ symbol.");
          return false;
        }
    }

    function validateEmail(email)
    {
        var atpos  = email.indexOf("@");   // The indexOf() method returns the position of the first occurrence of a specified value in a string. // Default value of start is 0
        //alert(atpos);
        var dotpos = email.lastIndexOf(".");  // The lastIndexOf() method returns the position of the last occurrence of a specified value in a string. // Default value of start is 0
        //alert(dotpos);

        if((atpos<1) || (dotpos<(atpos+2)) || (dotpos+2>=email.length))
```

```
    {
        return true;
    }
    else
    {
        return false;
    }
}

// Declaring required variables
var digits = "0123456789";
// non-digit characters which are allowed in phone numbers
var phoneNumberDelimiters = "()- ";
// characters which are allowed in international phone numbers
// (a leading + is OK)
var validWorldPhoneChars = phoneNumberDelimiters + "+";
// Minimum no of digits in an international phone no.
var minDigitsInIPhoneNumber = 10;

function isInteger(s)
{   var i;
    for (i = 0; i < s.length; i++)
    {
        // Check that current character is number.
        var c = s.charAt(i);
        if (((c < "0") || (c > "9"))) return false;
    }
    // All characters are numbers.
    return true;
}

function trim(s)
{   var i;
    var returnString = "";
```

```
    // Search through string's characters one by one.
    // If character is not a whitespace, append to returnString.
    for (i = 0; i < s.length; i++)
    {
       // Check that current character isn't whitespace.
       var c = s.charAt(i);
       if (c != " ") returnString += c;
    }
    return returnString;
}

function stripCharsInBag(s, bag)
{   var i;
    var returnString = "";
    // Search through string's characters one by one.
    // If character is not in bag, append to returnString.
    for (i = 0; i < s.length; i++)
    {
       // Check that current character isn't whitespace.
       var c = s.charAt(i);
       if (bag.indexOf(c) == -1) returnString += c;
    }
    return returnString;
}

function checkInternationalPhone(strPhone){
    var bracket=3;
    strPhone=trim(strPhone);
    if(strPhone.indexOf("+")>1) return false;
    if(strPhone.indexOf("-")!=-1)bracket=bracket+1;
    if(strPhone.indexOf("(")!=-1 &&
strPhone.indexOf("(")>bracket)return false;
    var brchr=strPhone.indexOf("(");
```

```
        if(strPhone.indexOf("(")!=-1 &&
strPhone.charAt(brchr+2)!=")")return false;
        if(strPhone.indexOf("(")==-1 && strPhone.indexOf(")")!=-1)return
false;
        s=stripCharsInBag(strPhone,validWorldPhoneChars);
        return (isInteger(s) && s.length >= minDigitsInIPhoneNumber);
    }
</script>
<?php
require_once "healthhelper.php";
$helper = new HealthHelper();

$db = new Database();
$db->open();

$msg = "";
if($_POST)
{
    $r = $helper->addAppointment();
    if($r)
    {
        $msg='<div class="alert alert-info" role="alert">
                                <h3>Your Appointment Added
Successfully.</h3>
                            </div>';
    }
    else
    {
        $msg='<div class="alert alert-info" role="alert">
                                <h3>Your Appointment Not Added
Successfully.</h3>
                            </div>';
    }
}
```

```
?>
</head>
<body class="boxed"  style="background-image:
url('images/backgrounds/images/img3.jpg'); background-repeat: no-repeat;
background-size: cover;">
<!--[if lt IE 7]>
      <p class="chromeframe">You are using an outdated browser. <a
href="http://browsehappy.com/">Upgrade your browser today</a> or <a
href="http://www.google.com/chromeframe/?redirect=true">install Google
Chrome Frame</a> to better experience this site.</p>
<![endif]-->
<!-- Preloader -->
<div id="preloader">
  <div id="status"></div>
</div>
<!-- Start Body Container -->
<div class="body footer-style2">
  <!-- Start Header -->
  <?php
   require_once "header.php";
  ?>
  <!-- End Header -->
  <!-- Start Content -->
  <div class="main" role="main">
    <div id="content" class="content page-content full">
     <header class="page-header flexible parallax text-align-center parallax-
overlay" style="background-image:url(images/img4.jpg)">
       <section>
         <div class="container">
           <div class="row">
             <div class="col-md-12">
               <h1>Book Appointment</h1>
             </div>
```

```
          </div>
        </div>
      </section>
    </header>
    <div class="container">
      <div class="row">
        <div class="col-md-12">
          <h2><strong>Book Appointment</strong></h2>
          <hr/>
          <p><?php echo $msg;?></p>
          <form method="post" action="" onsubmit="return validate_form();"
enctype="multipart/form-data">
              <div class="row">
                <div class="col-md-4">
                  <div class="form-group">
                  <label>Date</label>
                  <input type="date" id="event_date" name="event_date"
class="form-control input-lg" placeholder="Event Date" min="<?php echo
date('Y-m-d'); ?>" max="2045-12-31" />
                  </div>
                </div>
                <div class="col-md-4">
                  <div class="form-group">
                  <label>Name</label>
                  <input type="text" id="name" name="name"  class="form-
control input-md" placeholder="Name"/>
                  </div>
                </div>
                <div class="col-md-4">
                  <div class="form-group">
                    <label>Phone</label>
                    <input type="text" id="phone" name="phone"
class="form-control input-md" maxlength="10" placeholder="Phone"/>
                  </div>
```

```
                </div>
                <div class="col-md-4">
                    <div class="form-group">
                        <label>Email</label>
                        <input type="text" id="email" name="email"
class="form-control input-md" placeholder="Email"/>
                    </div>
                </div>
            </div>

            <div class="row">
                <div class="form-group">
                    <div class="col-md-12">
                        <input type="hidden" id="user_id" name="user_id"
value="<?php echo $_GET['user_id']; ?>" />
                        <input type="hidden" id="disease_predicted"
name="disease_predicted" value="<?php echo $_GET['disease_predicted'];
?>" />
                        <input type="submit" name="submit" class="btn btn-
primary btn-lg" value="Submit"/>
                    </div>
                </div>
            </div>
          </form>
        </div>
      </div>
    </div>
  </div>
  <?php
    require_once "footer.php";
  ?>
</div>
<!-- End Body Container -->
```

```
<script src="js/jquery-latest.min.js"></script> <!-- Jquery Library Call -->
<script src="plugins/prettyphoto/js/prettyphoto.js"></script>
<script src="plugins/prettyphoto/js/prettyphoto.js"></script>
<script src="plugins/owl-carousel/js/owl.carousel.min.js"></script>
<script src="plugins/page-scroller/jquery.pagescroller.js"></script>
<script src="js/helper-plugins.js"></script> <!-- Plugins -->
<script src="js/bootstrap.js"></script> <!-- UI -->
<script src="js/init.js"></script> <!-- All Scripts -->
<!-- Preloader -->
<script type="text/javascript">
    //<![CDATA[
        $(window).load(function() { // makes sure the whole site is loaded
            $("#status").fadeOut(); // will first fade out the loading animation
            $("#preloader").delay(350).fadeOut("slow"); // will fade out the white DIV that covers the website.
        });
    //]]>
</script>
<!-- End Js -->
</body>
</html>
```

Books: -

[1] “Web Programming”, by ‘Chris Bates’ Wiley Dreamtech India, 2nd Edition.

[2] “Software Engineering”, Ian Somerville, Sixth Edition, Pearson Education Ltd.

[3] “HTML Complete References” Easy steps to develop web pages.

[4] “PHP Complete Reference”

Websites: -

[1] http://en.wikipedia.org/wiki/PHP for Php.

[2] http://www.hotscripts.com/category/php/ for Php

[3] http://www.mysql.com/click.php?e=35050 for MySQL.

[4] http://www.w3schools.com for information on HTML.

Appendices

Chapter 1

Chapter 2

List of Figures:-

List of Tables:-