Course Code: CSE-434
Course Title: Software Engineering (Sessional)
Project Name: Doctor, Chamber and patient Management System (web based)

Team Members

Souhardyo Bhattacharjee
ID: 1804079

Md Nahid Imtiaz
ID: 18040101

Abu Saiyed Mohammad Sadat
ID: 1804105

Supervised By

Mir Md. Saki Kowsar
Assistant Professor
Department Of Computer Science and Engineering
Chittagong University of Engineering and Technology

Moumita Sen Sarma
Lecturer
Department Of Computer Science and Engineering
Chittagong University of Engineering and Technology

Department Of Computer Science and Engineering
Chittagong University of Engineering and Technology
Bangladesh
November, 2023

Contents

List of Figures

1 Executive Summary

The project focuses on transforming healthcare delivery through the enhancement of doctor-patient management. By prioritizing effective communication, cultural competence training, and the integration of health information technology, the project aims to promote patient-centered care and reduce healthcare disparities. Emphasis is placed on fostering stronger relationships between doctors and patients, empowering patients through shared decision-making, and ensuring equitable access to quality healthcare services. Through collaborative efforts and continuous professional development, the project seeks to create a healthcare environment characterized by empathy, inclusivity, and excellence in patient care.

This comprehensive project aims to revolutionize healthcare delivery by improving doctor-patient management through various strategic initiatives. It emphasizes enhancing communication channels, such as telemedicine platforms, to facilitate convenient and timely interactions between doctors and patients. Additionally, cultural competence training for healthcare providers is prioritized to ensure that diverse patient populations receive equitable and sensitive care. The project also focuses on leveraging health information technology, such as electronic health records and data analytics, to streamline clinical workflows and enhance diagnostic accuracy. Moreover, promoting patient engagement and shared decision-making empowers patients to actively participate in their healthcare journey, leading to better treatment adherence and outcomes. By addressing these key areas, the project seeks to foster a patient-centered approach to healthcare delivery, ultimately improving patient satisfaction and reducing healthcare disparities.

2 Overview of the problem

Prescription creation is a critical aspect of healthcare, and the traditional manual process can be prone to errors, inefficiencies, and lack of standardization. A prescription-maker tool aims to address these challenges by providing a digital solution for healthcare professionals to generate accurate, standardized, and secure prescriptions.

3 Software Requirements Specification (SRS) of Prescription Making System.

3.1 Introduction

3.1.1 Purpose

The purpose of the prescription feature in the app is to facilitate seamless communication between healthcare professionals and patients, allowing doctors to prescribe medications digitally. This functionality aims to enhance accessibility, streamline the prescription process, and improve overall patient care by providing a secure and efficient platform for prescription management. Through the app, patients can easily access their prescribed medications, dosage instructions, and relevant health information, fostering a more connected and patient-centric healthcare experience.

3.1.2 Scope

The scope of the prescription encompasses the recommendation of a specific medication tailored to address the patient's diagnosed medical condition. It involves defining the precise dosage, frequency, and duration of the prescribed medication, taking into account the patient's medical history, allergies, and potential drug interactions. Additionally, the scope extends to the ongoing monitoring of the patient's response to the prescribed treatment, allowing for adjustments or modifications as needed to ensure optimal therapeutic outcomes.

3.2 Overall Description

3.2.1 Product Perspective

From a product perspective, the prescription feature within the app acts as a crucial component in bridging the gap between healthcare providers and patients, digitizing and simplifying the prescription process. It provides a user-friendly interface for doctors to generate accurate and secure digital prescriptions, ensuring efficiency and reducing errors. Additionally, patients gain a centralized platform to access and manage their prescriptions, fostering a more organized and transparent healthcare experience.

3.2.2 User Classes and Characteristics

- User: Patient.
- Admins: Doctor, Administration of App.

3.2.3 Operating Environment

The system will be web-based, and accessible through standard web browsers. It should support common web technologies and databases.

3.3 Functional Requirements

3.3.1 User Authentication

- Users must be able to log in using their registered username/email and password.
- The system shall implement secure password storage techniques.
- Failed login attempts shall be tracked, and users shall be temporarily locked out after a predefined number of unsuccessful attempts.

3.3.2 User Authorization and Roles

- The system shall support different user roles such as regular user and administrator.
- Admin can manage (list, add, edit, delete) members.

3.3.3 User Profile Management

- Users shall be able to edit their profiles, including changing their passwords and updating personal information.
- Admin shall have the ability to add/update their specialization information.

3.3.4 Prescribe Medicine and Test

- The system allows doctors to electronically prescribe medications for patients using a standardized, secure interface.
- Doctors select medication from a pre-populated database, and specify dosage, duration, and instructions.
- System generates a printable prescription document and electronically transmits it to the pharmacy and patient health record.

3.4 Non-functional Requirements

3.4.1 Performance

- The system should support a large number of simultaneous user registrations and logins.
- User authentication and authorization processes should be completed within 3 seconds.
- The system shall be accessible via the internet.
- The system shall be available 24/7.
- The system shall be secure and protect user data.
- The system shall be easy to use and navigate.
- The system shall be scalable to accommodate a growing number of users and appointments.

3.4.2 Security

- Passwords must be securely stored using encryption techniques.
- All user interactions must be transmitted over HTTPS.

3.4.3 Usability

- The user interface for registration and profile management should be intuitive and user-friendly.
- Error messages should be clear and helpful.

3.5 Constraints

- The system must be developed using the MEAN stack, including MongoDB, Express.js, Angular, and Node.js.
- The system must run on a server environment that supports Node.js.
- The system must be compatible with Bootstrap for responsive web design.
- The system must comply with relevant data protection regulations and standards.

3.6 Assumptions and Dependencies

- The system assumes that users have access to the internet and a web browser.

3.7 Future Enhancements

- Enhancement: Implement advanced analytics tools and reporting features to provide deeper insights into medicine stock trends, usage patterns, and forecasting.
- Improved decision-making for inventory management, reducing excess stock and minimizing shortages.
- Implement machine learning algorithms to predict medicine demand based on historical data, seasonal trends, and other relevant factors.
- Optimize stock levels and reduce the likelihood of stockouts or overstock situations.

3.8 Conclusion

In conclusion, the prescribed medication serves as a targeted therapeutic approach to address the patient's specific medical needs, taking into consideration factors such as the diagnosis, individual health profile, and any potential contraindications. The prescribed regimen, comprising specified dosage, frequency, and duration, aims to optimize therapeutic benefits while minimizing risks. The patient must adhere to the prescribed treatment plan and maintain open communication with their healthcare provider to monitor and address any potential side effects or adjustments needed for optimal health outcomes. This collaborative approach fosters a patient-centered and personalized care strategy.

4 Software Requirements Specification (SRS) of Appointment Management System.

4.1 Introduction

4.1.1 Purpose

This system aims to streamline and automate the process of managing appointments for medical appointments.

4.1.2 Scope

The Appointment Management System will allow users to schedule, modify, and cancel appointments efficiently. It will provide a user-friendly interface for both administrators and users, ensuring effective communication and organization of appointments.

4.2 Overall Description

4.2.1 Product Perspective

The Appointment Management System operates within a broader technological context, relying on various interfaces to deliver a seamless user experience. It interfaces with web browsers, ensuring compatibility with major platforms, and interacts with a relational database management system for efficient data storage and retrieval. Communication protocols such as HTTP/HTTPS, SMTP/SMTPS. The user interfaces are designed to be intuitive, with features like dashboards, schedulers, and administrative panels.

4.2.2 User Classes and Characteristics

- User: Any Patient.
- Admins:Doctor, Administration of App.

4.2.3 Operating Environment

The Appointment Management System is designed to operate in a versatile environment, accessible across standard hardware configurations, including desktops, laptops, tablets, and smartphones.

4.3 Functional Requirements

4.3.1 Appointment Creation:

- Users shall be able to schedule appointments by selecting a date, time, and type/category of the appointment.

4.3.2 Appointment Modification and Cancellation:

- Users should have the capability to modify or cancel their appointments within a defined time frame, with appropriate notifications sent to affected parties.

4.3.3 Search and Filter:

- Users must be able to search for available time slots based on specific criteria and filter appointments accordingly.

4.3.4 Appointment Reminders:

- The system shall send automated reminders to users before their scheduled appointments via email or SMS based on their specified preferences.

4.4 Non-functional Requirements

4.4.1 Performance

- The system shall be accessible via the internet.
- Inventory Update should be complete in 5 seconds.
- The system shall be available 24/7.

4.4.2 Security

- Inventory details must be securely stored using encryption techniques.
- Access to Inventory information should be restricted to authorized users.

4.4.3 Usability

- The user interface for Inventory management should be intuitive and user friendly.
- Notification messages should be clear and provide relevant details.

4.5 Constraints

- The system must be developed using the MERN stack, including MongoDB, Express.js, Angular, and Node.js.
- The system must run on a server environment that supports Node.js.
- The system must be compatible with Bootstrap for responsive web design.
- The system must comply with relevant data protection regulations and standards.

4.6 Assumptions and Dependencies

- The system assumes that users have access to the internet and a web browser.

4.7 Future Enhancements

- The system could be enhanced to allow patient to make appointment from their phones.
- The system could be enhanced to provide additional flexibility and convenience to Patients.
- The system could be enhanced to provide better reliability and security of information.

4.8 Conclusion

The Appointment Management System offers a flexible and user-friendly solution for efficient appointment scheduling and organization. Operating seamlessly across various devices and platforms, it ensures accessibility for a diverse user base. With compatibility across major web browsers and support for standard operating systems, the system provides a reliable and intuitive experience. Leveraging a robust database management system and secure network protocols, it prioritizes data integrity and user privacy. By adhering to regulatory compliance, the system demonstrates a commitment to safeguarding sensitive information. Overall, the system's adaptability and emphasis on user experience make it a valuable tool for streamlined appointment management in diverse operational environments.

5 Software Requirements Specification (SRS) of Payment Receive System

5.1 Introduction

5.1.1 Purpose

It efficiently collects payments for medical services, and maintains a comprehensive record of patients financial transactions. The system ensures accurate financial tracking, facilitates compliance with regulatory standards, and provides valuable data for audits. Moreover, it enables users to offer transparent billing information to patients, fostering trust and accountability in the financial aspects of healthcare delivery.

5.1.2 Scope

It providing a centralized platform for efficient payment processing. It streamlines revenue collection for various medical services, ensures accurate billing. This scope extends to comprehensive financial tracking, allowing users to maintain transparent patient billing histories. The system facilitates compliance with regulatory standards, supports auditing processes, and enhances overall financial planning.

5.2 Overall Description

5.2.1 Product Perspective

It functions as a crucial component, enhancing the product perspective by streamlining revenue collection and financial tracking. It seamlessly integrates within the patient management system, providing a user-friendly and efficient solution for managing healthcare-related payments within a unified platform.

5.2.2 User Classes and Characteristics

- User: Patient
- Admin: Assistant

5.3 Functional Requirements

5.3.1 Security

- Passwords must be securely stored using encryption techniques.
- All user interactions must be transmitted over HTTPS.

5.4 Non-functional Requirements

5.4.1 Usability

- The interface should be user-friendly for both admin.

5.4.2 Performance

- The system should handle a high volume of data efficiently.
- The process should be completed securely and promptly.

5.5 Constraints

- The system must be developed using the MERN stack, including MongoDB, Express.js, Angular, and Node.js.
- The system must run on a server environment that supports Node.js.
- The system must be compatible with Bootstrap for responsive web design.
- The system must comply with relevant data protection regulations and standards.

5.6 Assumptions and Dependencies

- Admins assume that user provide valid input.

5.7 Future Enhancements

- The system could be enhanced to automated payment system without help of assistant.
- The system could be enhanced to provide better reliability and security.

5.8 Conclusion

In conclusion, the Payment Receive System is an indispensable element that significantly contributes to the efficiency and transparency of healthcare financial processes. By automating payment workflows, ensuring accurate billing, it streamlines revenue collection and financial tracking. This not only enhances the overall financial health of healthcare facilities but also fosters trust and transparency in patient-provider interactions. The system's ability to maintain comprehensive billing histories and support regulatory compliance further solidifies its role as an essential tool, ultimately enabling healthcare providers to focus on delivering quality care while maintaining a robust and organized approach to financial management.

6 Use Case Diagram

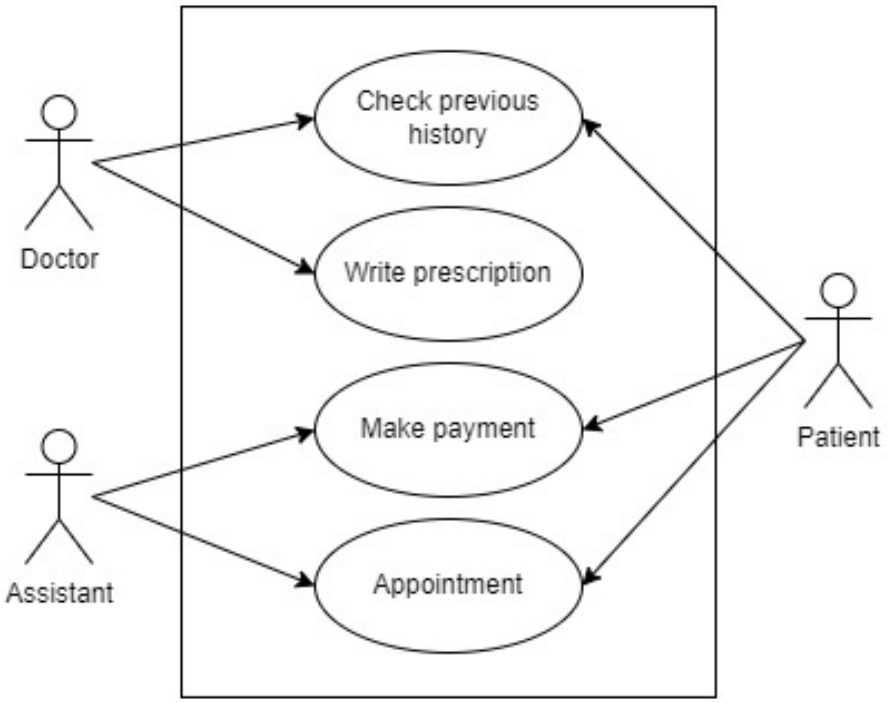

Figure 1: Use Case Diagram

Use Case:

Prescription Making System.

Iteration:

1

Primary Actor:

Doctor.

Goal in Context:

To prescribe medicine for the Patient.

Preconditions:

- System must be fully automated.
- Primary Actors have to access the system.

Trigger:

Users accessing the system for the first time.

Scenario:

1. The Doctor reviews the patient's previous records and test results, establishing a baseline for the follow-up.
2. The Doctor evaluates the patient's adherence to the prescribed medication and any observed effects.
3. Patient shares updates on symptoms and any concerns, fostering open communication.
4. The Doctor modifies the treatment plan based on the review, addressing persisting issues or side effects.
5. Doctor and patient collaboratively schedule follow-up tests or consultations for ongoing monitoring and adjustments.

Exceptions:

- Invalid registration information - prompt for correct details.
- Unsuccessful registration confirmation - resend confirmation email or contact support.
- Incorrect login credentials - prompt for correct information.

Priority:

High priority, as it is fundamental to user interaction with the system.

When Available:

First increment

Frequency of Use:

Frequent

Channel to Actor:

Via a web browser on a PC or mobile device using an internet connection.

Secondary Actors:

Patient

Channels to Secondary Actors:

Via a web browser on a PC or mobile device using an internet connection.

Open Issues:

- What measures are in place to ensure the security of user registration information during the confirmation process?
- How will the system handle forgotten passwords and account recovery?

6.1 Use Case Diagram for Prescription Making System.

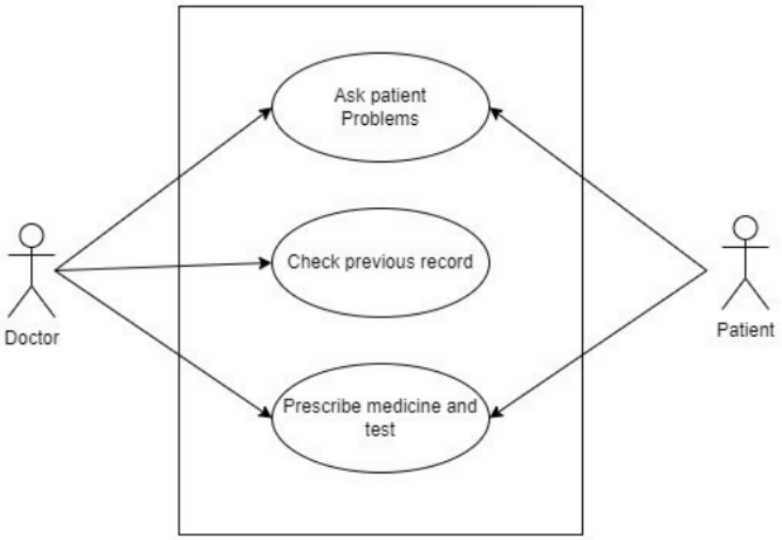

Figure 2: Use Case Diagram for Prescription Making System.

6.1.1 Activity Diagram for Prescription Making System.

The activity diagram for the Prescription Making System begins with the doctor logging into the system. Once logged in, the doctor can view previous patient records, facilitating an informed decision-making process. The doctor then engages in the prescription creation activity, specifying medication details, dosage, and instructions. Subsequently, the doctor logs out, ensuring the security of patient information. On the patient's side, the activity starts with logging into the system, followed by accessing the prescribed medication details. Finally, the patient logs out, concluding the interaction with the prescription system.

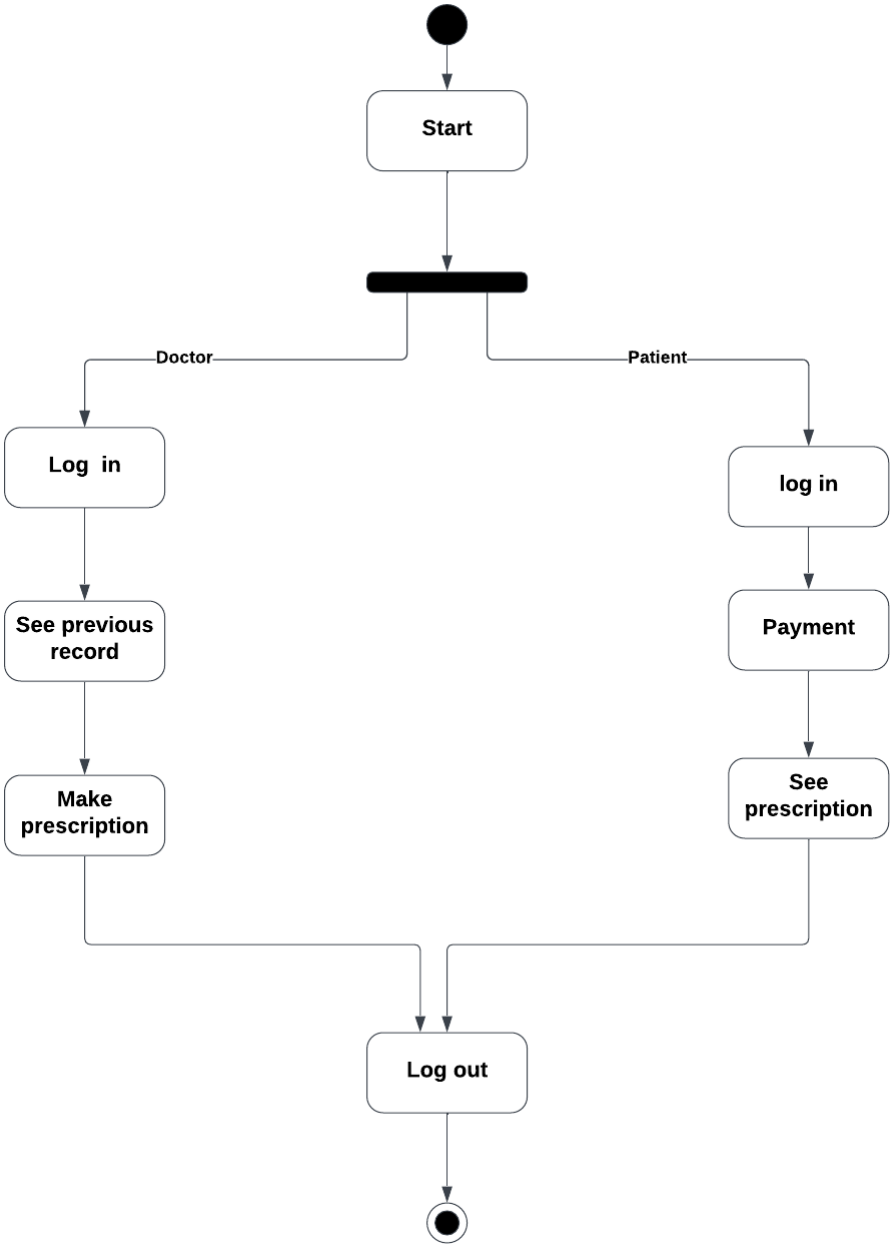

Figure 3: Activity Diagram for Prescription Making System.

6.1.2 Static Model – Class Diagram for Prescription Making System.

Classes: Patient:

- Attributes: Name, Phone Number, Email Address
- Operations: Take Treatment()

Doctor:

- Attributes: Name, Email, Location
- Operations: Check Previous History(), Check Test Result(), Make Prescription()

Old Patient: (inherits from Patient)

- Attributes: Patient Number, Patient Name
- Operations: Take Treatment()

New Patient: (inherits from Patient)

- Attributes: Patient Number, Patient Name
- Operations: Take Treatment()

Prescription:

- Attributes: Name, Dosage, Start Date, End Date

Relationships:
Patient to Doctor:

- "Take Treatment" (0.1 to 1)
- A patient can take treatment from 0 or 1 doctor.

Doctor to Patient:

- "Has" (1 to 1)
- A doctor has 1 patient at a time.

Doctor to Prescription:

- "Make" (1 to 1)
- A doctor makes 1 prescription.

Patient to Prescription:

- "Take" (1 to 1)
- A patient takes 1 prescription.

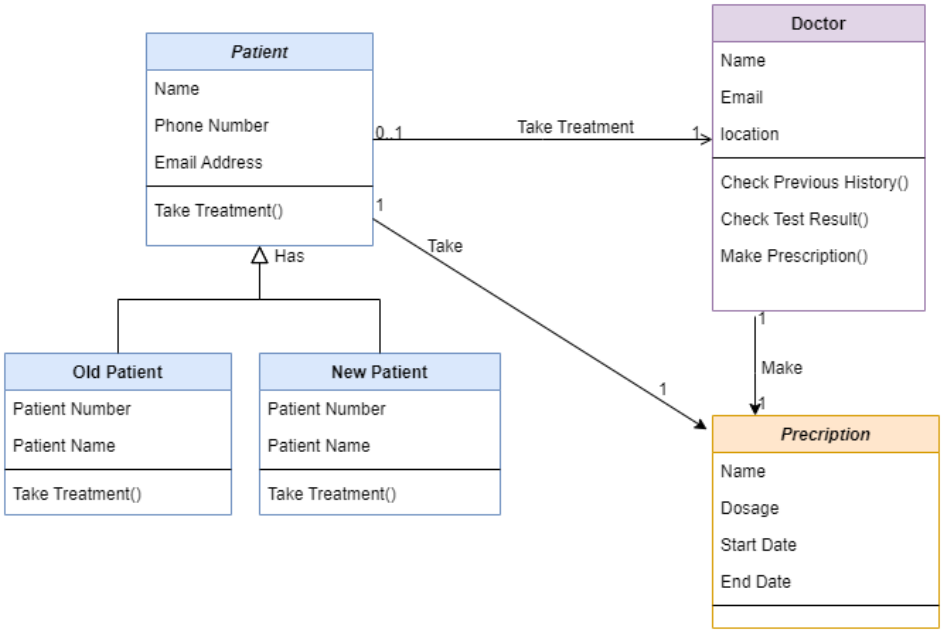

Figure 4: Class Diagram for Prescription Making System.

6.1.3 Dynamic Model – Sequence Diagram for Prescription Making System.

This sequence diagram outlines a prescription making process involving a doctor, patient, prescription, and database. The doctor asks the patient about their problems. After receiving the response, the doctor requests previous records from the database. The database responds with the records. Using this information, the doctor prescribes medicine and tests to the patient and saves this prescription in the database.

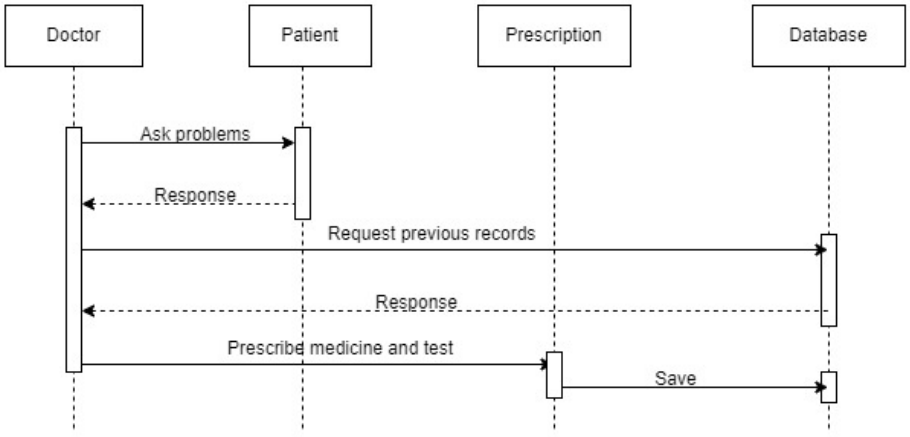

Figure 5: Sequence Diagram for Prescription Making System.

Use Case:

Appointment Management System.

Iteration:

2

Primary Actor:

Patient.

Goal in Context:

The aims to streamline and automate the process of managing appointments for medical appointments.

Preconditions:

- The system must be fully configured.
- The user must have a registered account with the appropriate user ID and password.

Trigger:

The patient can make ,modify or cancel appointment.

Scenario:

1. Users make appointment in the system securely.
2. Users navigates to inventory database.

Exceptions:

- If a user attempts to cancel an appointment outside the allowed time frame, inform them of the policy and, if necessary, provide contact information for assistance
- Invalid login credentials - prompt for correct information.
- Failure to submit the service request - display error message; check the internet connection.

Priority:

High priority, fundamental to the system's purpose.

When Available:

first increment

Frequency of Use:

Frequent, as users encounter issues or concerns.

Channel to Actor:

Via a web browser on a PC or mobile device using an internet connection.

Secondary Actors:

Assistant

Channels to Secondary Actors:

Via a web browser on a PC or mobile device using an internet connection.

Open Issues:

- What specific mechanisms are in place to ensure the security and privacy of data related to user complaints?

6.2 Use Case Diagram for Appointment Management System.

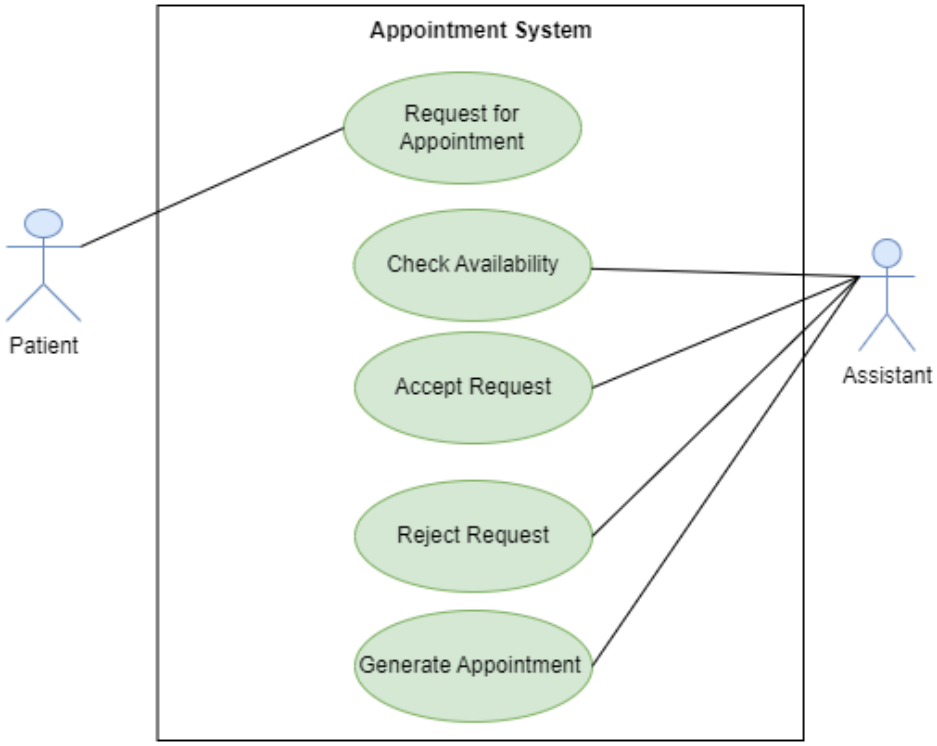

Figure 6: Use Case Diagram for Appointment Management System.

6.2.1 Activity Diagram for Appointment Management System.

In the Activity Diagram for the Appointment Management System, the patient initiates the process by logging into the system. After logging in, the patient proceeds to make a payment for the appointment, ensuring a seamless financial transaction. The next activity involves fixing a date and time for the appointment, allowing the patient to choose a convenient schedule. Finally, the patient logs out, completing the interaction with the Appointment Management System.

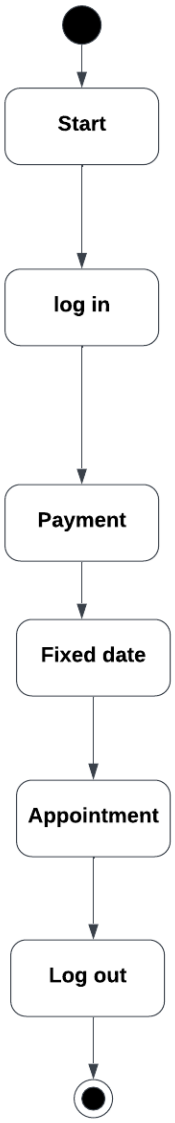

Figure 7: Activity Diagram for Appointment Management System.

6.2.2 Static Model – Class Diagram for Appointment Management System.

Classes: Patient:

- Attributes: Name, Phone Number, Email Address
- Operations: Request for Appointment(): Enables a patient to request an appointment.

Assistant:

- Attributes: Name, Phone Number, Email Address
- Operations:Receive Request(): Allows the assistant to receive appointment requests.
- Operations:Accept or Reject Request(): Enables the assistant to either accept or reject a request.
- Operations:Generate Appointment(): Creates an appointment if a request is accepted.

Old Patient: (inherits from Patient)

- Attributes: Name, Phone Number, Email Address
- Operations:Request for Appointment(): Enables a patient to request an appointment.

New Patient: (inherits from Patient)

- Attributes: Name, Phone Number, Email Address
- Operations:Request for Appointment(): Enables a patient to request an appointment.

Relationships:
Association:

- An association between Patient and Assistant, indicating that a patient has an assistant to interact with

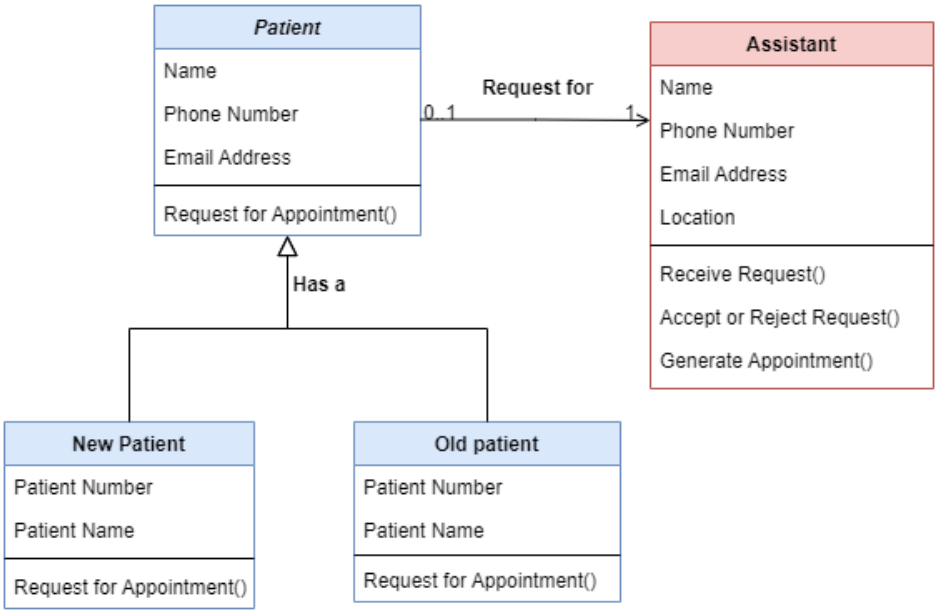

Figure 8: Class Diagram for Appointment Management System.

6.2.3 Dynamic Model – Sequence Diagram for Appointment Management System.

This sequence diagram outlines the process of a patient making an appointment. The patient requesting for appointment, and the assistant checks the database for availability. If available, request will be accepted and an appointment will be generated; if not, there's an alternative path where the request will be rejected.

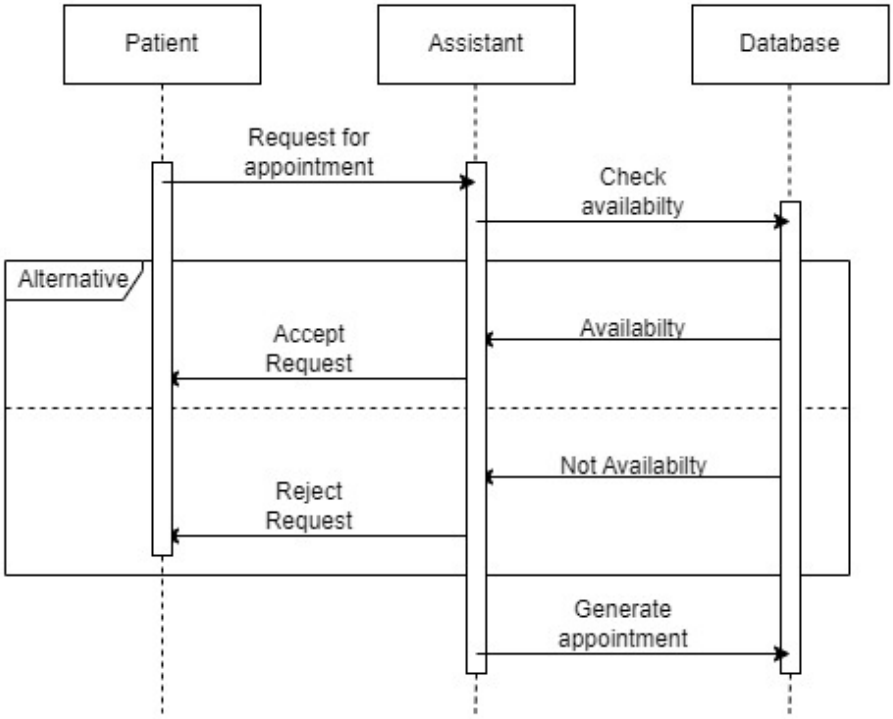

Figure 9: Sequence Diagram for Appointment Management System.

Use Case:

Payment Receive System.

Iteration:

3

Primary Actors:

Assistant

Goal in Context:

The assistant receive payment from the patient and save payment record in the system.

Preconditions:

- Assistant must has a registered account.
- Patient has to consult with doctor.

Trigger:

The assistant receive payment from patient.

Scenario:

1. The assistant receive payment from patient.
2. The assistant save payment record in the system.

Exceptions:

- Invalid payment information - prompt for correct information.

Priority:

Low priority

When Available:

Second increment

Frequency of Use:

Frequent

Channel to Actors:

Via a web browser on a PC or mobile device using an internet connection.

Secondary Actors:

Patient

Channels to Secondary Actors:

Accessed through a desktop web browser using an internet connection.

Open Issues:

- How will the system handle and notify users of changes?

6.3 Use Case Diagram for Payment Receive System.

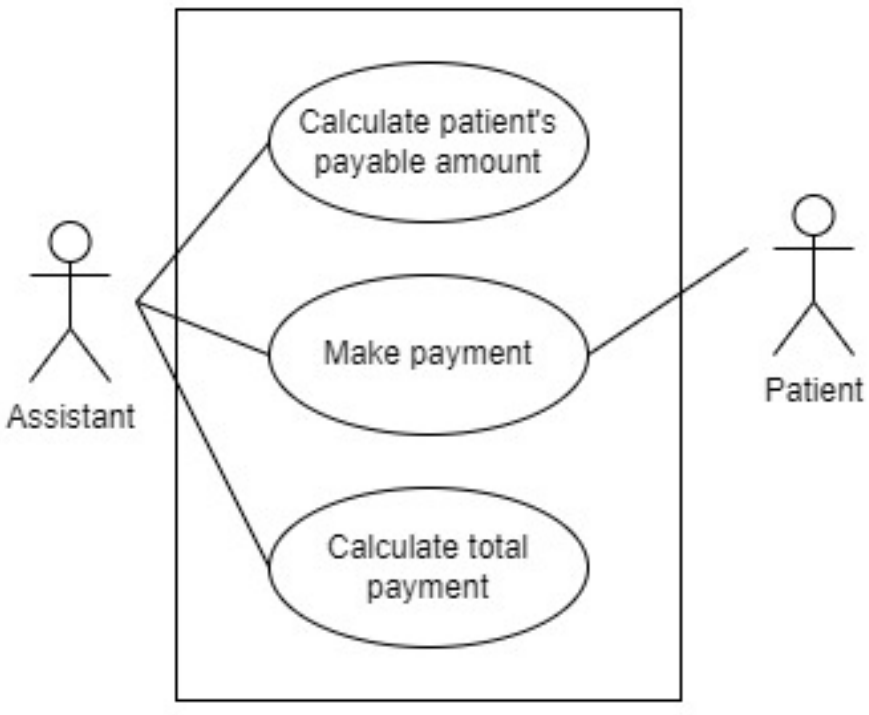

Figure 10: Use Case Diagram for Payment Receive System.

6.3.1 Activity Diagram for Payment Receive System.

In the Activity Diagram for the Payment Receive System, the patient begins by logging into the system to initiate a payment transaction. After logging in, the system processes the payment, capturing and confirming the financial transaction securely. Once the payment is successfully completed, the patient logs out, concluding the payment process within the Payment Receive System. This streamlined activity ensures a secure and efficient experience for users in managing their financial transactions.

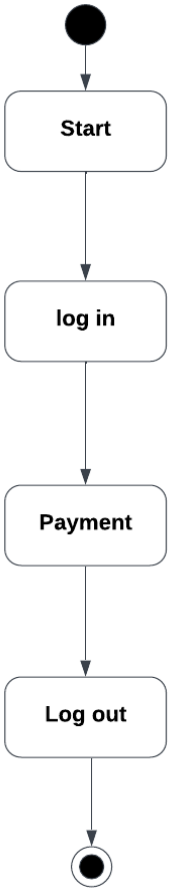

Figure 11: Activity Diagram for Payment Receive System.

6.3.2 Static Model – Class Diagram for Payment Receive System.

Classes: Patient:

- Attributes: Name, Phone Number, Email Address
- Operations: Make Payment()

Assistant:

- Attributes: Name, Phone Number, Email Address
- Operations: Receive Payment()

Old Patient: (inherits from Patient)

- Attributes: Name, Phone Number, Email Address
- Operations: Make Payment()

New Patient: (inherits from Patient)

- Attributes: Name, Phone Number, Email Address
- Operations: Make Payment()

Relationships:
Association:

- "Give Payment" between Patient and Assistant (1 to 1 cardinality)

Generalization:

- "Has" between Patient and New Patient/Old Patient (inheritance)

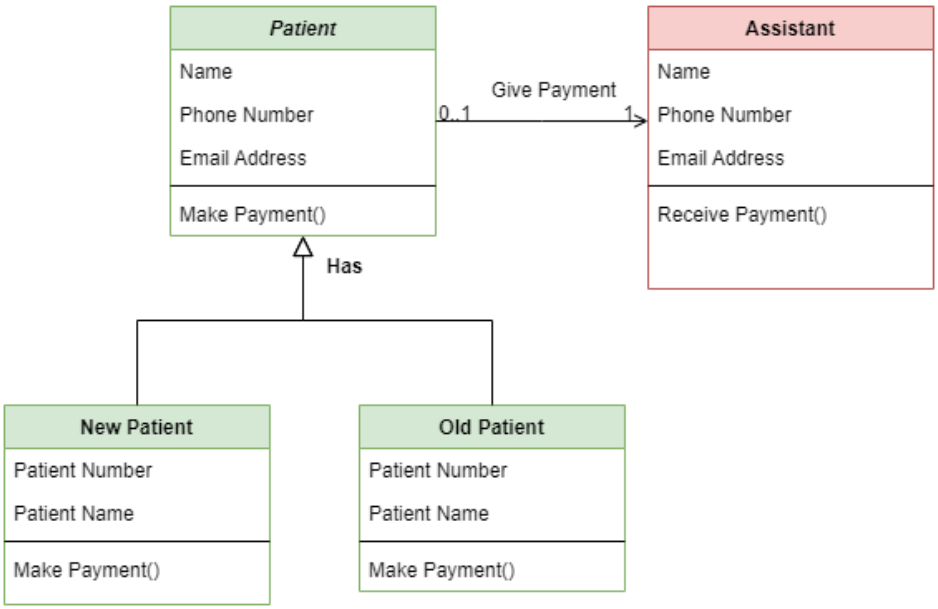

Figure 12: Class Diagram for Payment Receive System.

6.3.3 Dynamic Model – Sequence Diagram for Payment Receive System.

This sequence diagram outlines a payment process involving three entities: Assistance, Patient, and Database. Assistance asks the Patient for payment and then the Patient then makes the payment. Following this, Assistance saves the payment information in the Database.

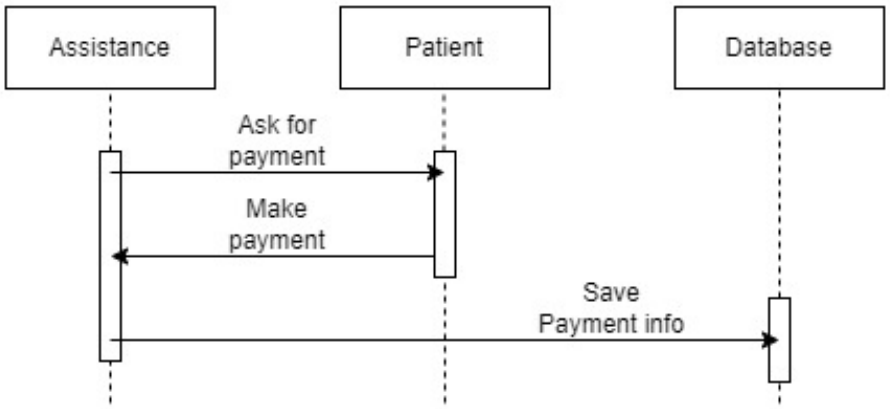

Figure 13: Sequence Diagram for Payment Receive System.

7 Safety and Security requirements

Ensuring the safety and security of our systems is paramount in safeguarding what holds significance for us. We've recognized and implemented measures to mitigate the risks posed by potential attacks exploiting system vulnerabilities. This proactive approach reduces the likelihood of issues arising and enhances the overall resilience of our system.

7.1 Access Requirements

Access requirements are crucial for ensuring that the right individuals have appropriate access to the features and data within the home management system while preventing unauthorized access.Some requirements includes user authentication,user roles and permission and password policies.

7.2 Integrity Requirements

Integrity requirements in the context of a home management system are essential to ensure the accuracy, reliability, and consistency of data and system processes. To ensure data integrity we have follow the following things.

- Implement mechanisms to prevent unauthorized or unintentional modification of data.
- Ensure that transactions, such as updates or modifications to the system, are processed reliably and completely.

7.3 Privacy Requirement

Privacy requirements are essential to safeguard the personal information of individuals using a home management system.To ensure privacy we have to do following things.

- Collect only the minimum amount of personal information necessary for the intended purpose.
- Obtain explicit and informed consent from users before collecting or processing their personal information.

7.4 MVC

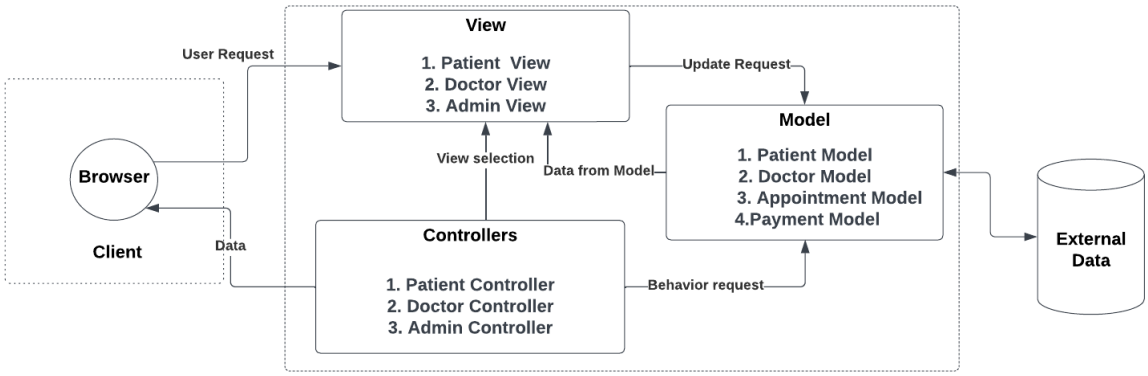

Figure 14: MVC

The Prescription Maker's Model-View-Controller (MVC) diagram consists of three main components:

7.4.1 Model (M):

The model represents the core data and business logic of the Prescription Maker, encompassing information such as patient details, medications, and dosage instructions. It ensures data integrity and manages the application's state.

7.4.2 View (V):

The view is responsible for presenting the user interface to the healthcare professional or user. It displays information from the model and allows users to interact with the system, including entering patient data and selecting medications.

7.4.3 Controller (C):

The controller acts as an intermediary between the model and view, handling user input and updating the model accordingly. It processes commands from the user interface, orchestrates data flow, and communicates changes between the model and view components.

7.4.4 User Interface (UI):

This component represents the graphical interface through which healthcare professionals interact with the Prescription Maker. It includes forms for entering patient information, selecting medications, and specifying dosage details.

7.5 Data Centered Architecture

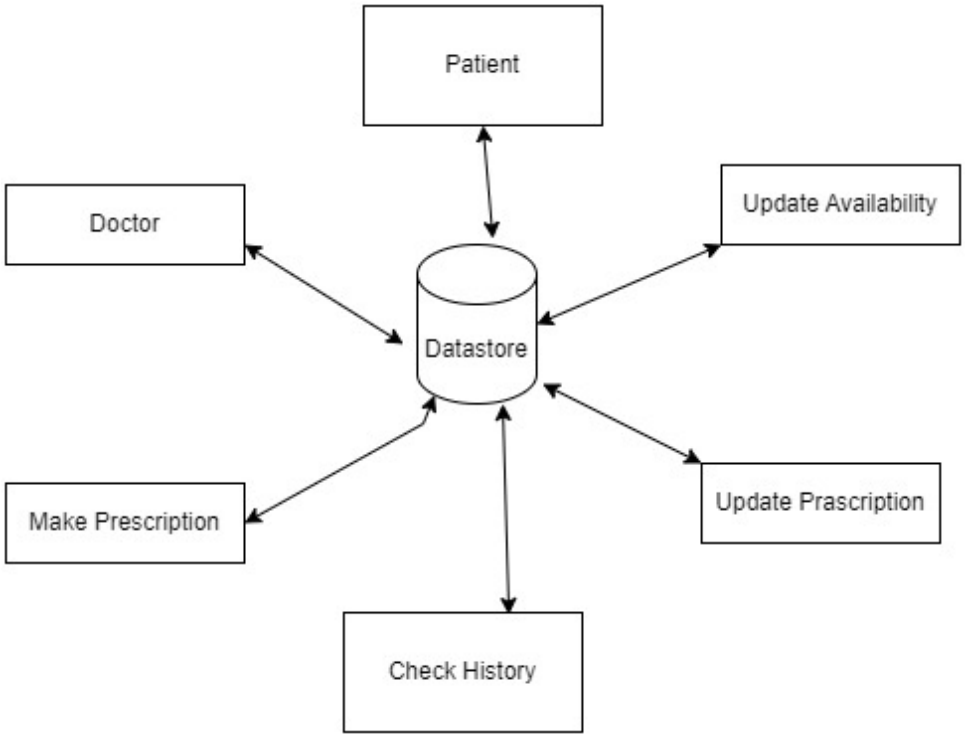

Figure 15: Data Centered Architecture

7.6 User Interface

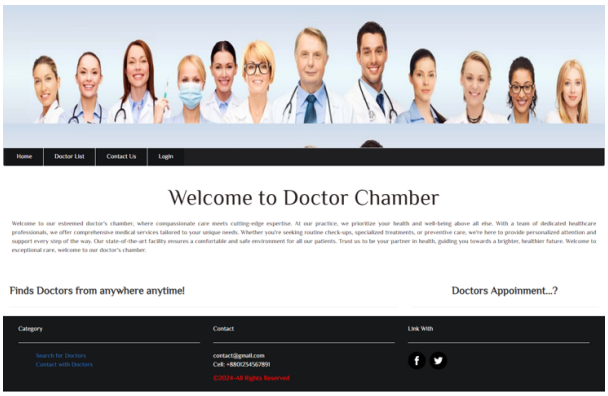

Figure 16: Home page

This is welcome page for the system.the user can enter here and see the welcome page.

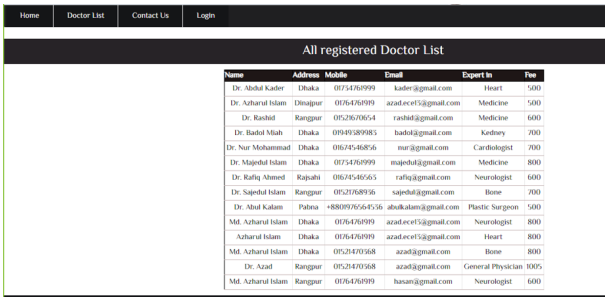

Figure 17: Doctor list

Here user can see the list of doctor of this chamber without sign in.

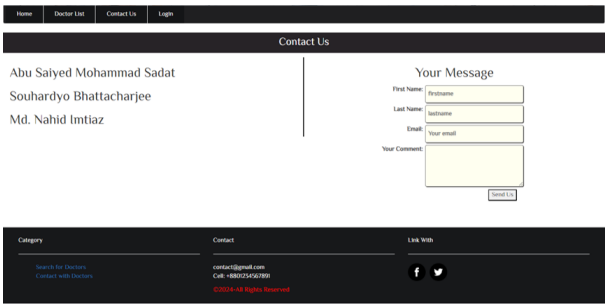

Figure 18: Contact us page

In this page user can contact with administration without sign in by providing their email, phone number and other details.

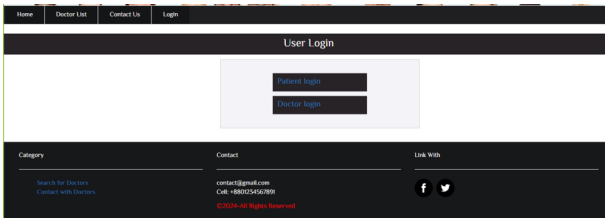

Figure 19: User type selection

From this page when a user want to sign in he can select what kind of user he is like doctor or patient. By selecting his role he will be redirecting to his desire login page.

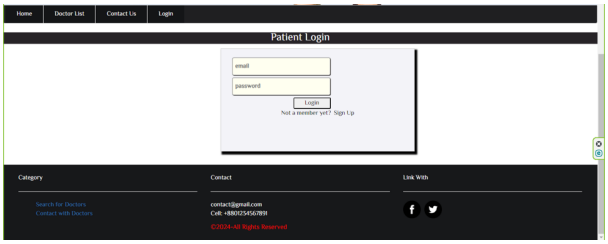

Figure 20: Patient log-in

Here patients can log in by entering their email and password to access their accounts. Below the password field, there's a "Forgot Password" option. A "SIGN IN" for sign in and a "SIGN UP" text button for going to sign up page.

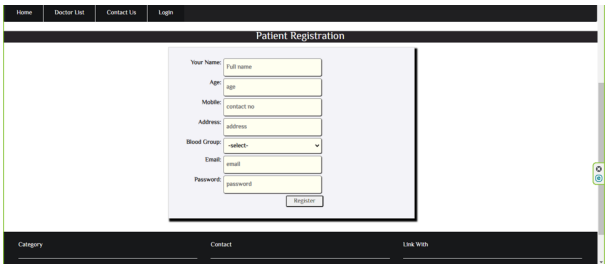

Figure 21: Patient registration

Here patients can create an account by entering their name, email, and create a password to sign up. A "SIGN UP" button for sign up and a "SIGN IN" text button for going to sign in page if user already having an account.

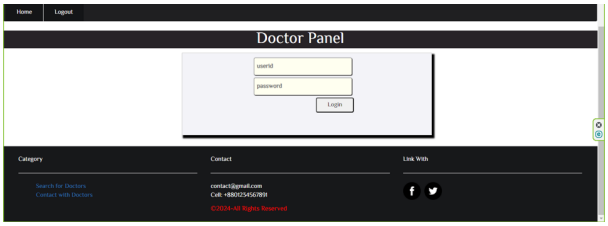

Figure 22: Doctor log-in

Here doctors can log in by entering their email and password to access their accounts. Below the password field, there's a "Forgot Password" option. A "SIGN IN" for sign in and a "SIGN UP" text button for going to sign up page.

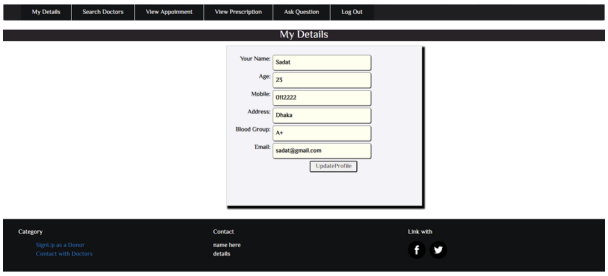

Figure 23: Patient details

This is patient details and profile edit page of this system which allows patients to view and modify their personal information. The form includes fields for name, age, mobile number, address, blood group, and email. Patients can update their profile by clicking the "Update Profile" button.

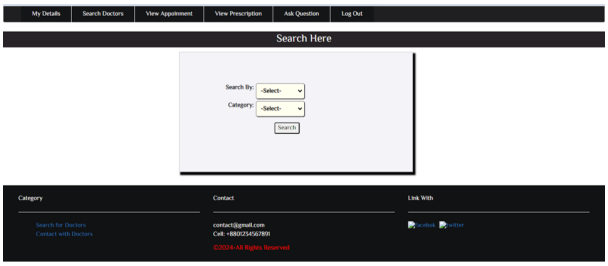

Figure 24: Doctor search

This is Doctor Search Page in this system which allows patients to find doctors efficiently. Users can filter doctors by selecting criteria from drop-down menus, such as specialty, location, and availability. The page provides a straightforward interface for patients seeking medical assistance.

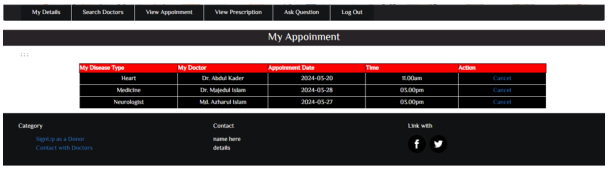

Figure 25: Check appointment

This Check Appointment Page of this system allows patients to view their upcoming appointments. It displays booked appointments with details such as the doctor's name, appointment date, and time. Patients can easily track their scheduled visits and manage their healthcare. The page also provides options to cancel appointments. The straightforward layout ensures efficient appointment management, enhancing patient experience and organization.

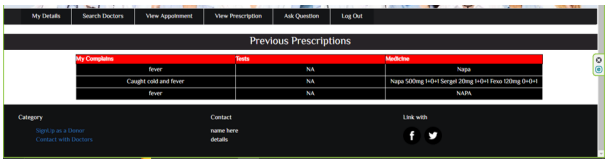

Figure 26: Previous prescription and history

The Previous Prescription Page of this system displays a record of past prescriptions for patients. It includes details such as the doctor's name, prescription date, medication, and dosage. Patients can refer to this page to track their medical history and ensure consistent treatment. The organized layout simplifies access to essential information, promoting efficient healthcare management.

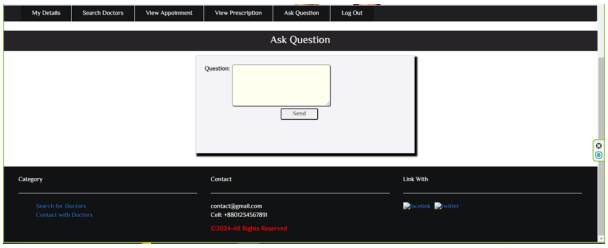

Figure 27: Asking question

The "Ask Question" page in the doctor chamber management system allows patients to interact with the doctor's assistant. Patients can type their health-related queries in the provided text box and submit them. The page aims to enhance communication and streamline healthcare services by facilitating patient inquiries and responses from the assistant.

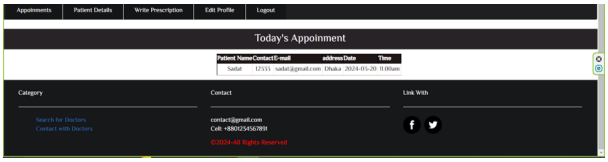

Figure 28: Appointment checking

The "Checking Appointments" page in the doctor chamber management system serves as an efficient tool for tracking scheduled appointments. It displays essential details about upcoming appointments, including the patient's name, contact email, address, and the appointment date and time. The interface is straightforward, allowing easy access to appointment information.

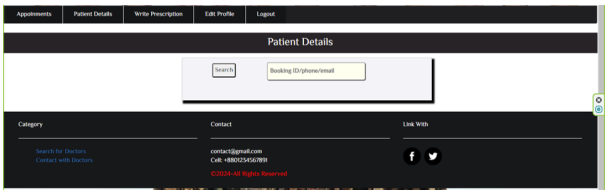

Figure 29: Patient details

The "Checking Patient Details" page in the doctor chamber management system allows users to efficiently retrieve patient information. Users can search for details using a Booking ID, phone number, or email. The interface is clean and organized, displaying essential patient data.

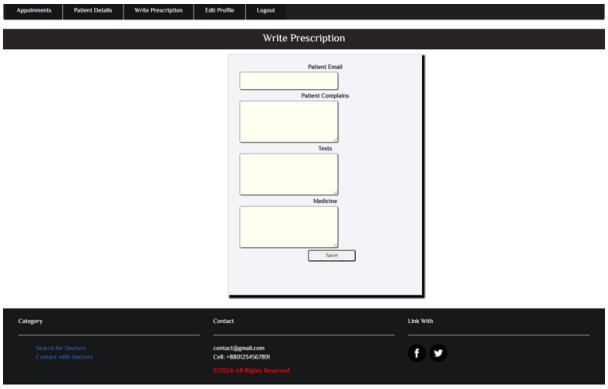

Figure 30: Prescription maker

A "prescription" section in a doctor-patient system serves as a critical component for healthcare professionals to create, manage, and share prescriptions with patients. This section involves various features to ensure accurate and secure handling of medication-related information.

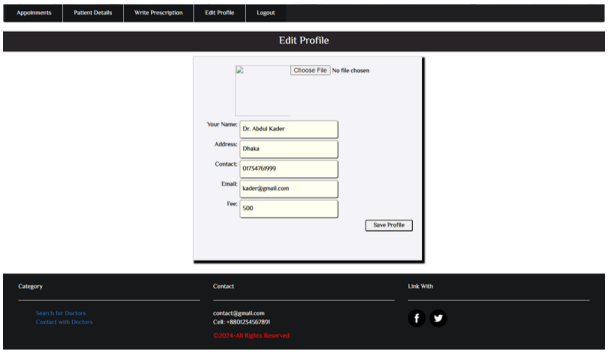

Figure 31: Editing profile

The "Edit Profile" page in the doctor chamber management system allows doctors to conveniently update their professional information. On this straightforward interface, doctors can modify their name, address, contact number, email, and fees. Additionally, there's an option to upload a profile picture. The page streamlines the process of maintaining accurate and up-to-date details for healthcare professionals, ensuring efficient communication and seamless management of their profiles.

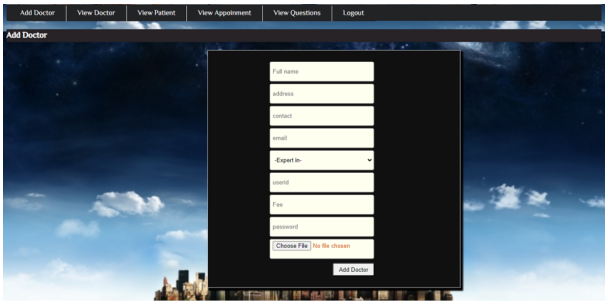

Figure 32: Add doctor

The "Add Doctor" page in the doctor chamber management system allows administrators to efficiently input and manage doctor information. The interface features a form where administrators can enter details such as full name, address, contact number, email, expertise area, user ID, fee, and even upload profile picture.This page streamlines the process of adding new doctors to the system, ensuring accurate and organized data for healthcare professionals.

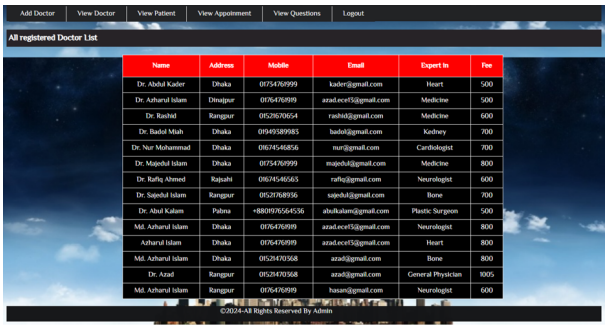

Figure 33: Doctor list

This webpage displays a doctor list within a chamber management system. It includes essential details of registered doctors, such as their names, addresses, mobile numbers, email addresses, areas of expertise, and fees. The page also provides options for adding doctors and viewing patients and appointments. Overall, it serves as a comprehensive resource for patients seeking medical assistance.

Figure 34: patient list

This webpage displays a patient list within a doctor's chamber management system. It includes essential details of registered patients, such as their patient IDs, names, ages, mobile numbers, addresses, blood groups, and email addresses. The page serves as an organized repository of patient information, facilitating easy access and efficient management for medical professionals.

Figure 35: Appointment list

This digital appointment list within a doctor's chamber management system meticulously organizes scheduled appointments. It displays crucial details, including the doctor's name, contact information, expertise, patient's name, contact, and the date and time of each appointment. The system streamlines appointment management, ensuring efficient scheduling and seamless patient care.

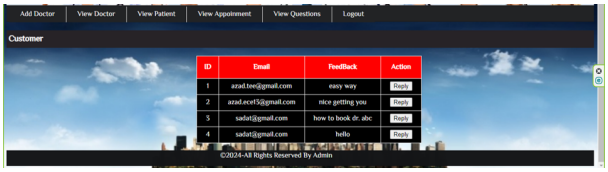

Figure 36: Asked question list

This webpage serves as a platform for doctor's assistants to address patient inquiries within a chamber management system. It displays a table labeled 'Customer' containing columns for ID, Email, Question, and Action. The page streamlines communication, allowing assistants to promptly respond to patients' questions and concerns, enhancing overall patient care and satisfaction.

7.7 Data Flow Diagram

7.7.1 Zero Level DFD For Systems

The Doctor-Patient Management System serves as the main process, coordinating various functionalities to facilitate efficient management of doctor-patient interactions. This system involves several key processes and entities: The process is responsible for handling appointment-related tasks. Manages the scheduling, modification, and cancellation of appointments.Interacts with the "Appointments" data store.Process focused on the management of prescriptions.Handles the creation, updating, and retrieval of patient prescriptions.Interacts with the "Prescriptions" data store.Process dealing with patientqueries and communications.Manages the submission, response, and tracking of queries.Interacts with the "Query Data" data store.Process responsible for handling financial transactions.Manages payment data, processing transactions related to medical services.Interacts with the "Payment Data" data store.Process facilitating user authentication and registration.Handles user sign-in and sign-up activities.Interacts with the "User Accounts" data store.

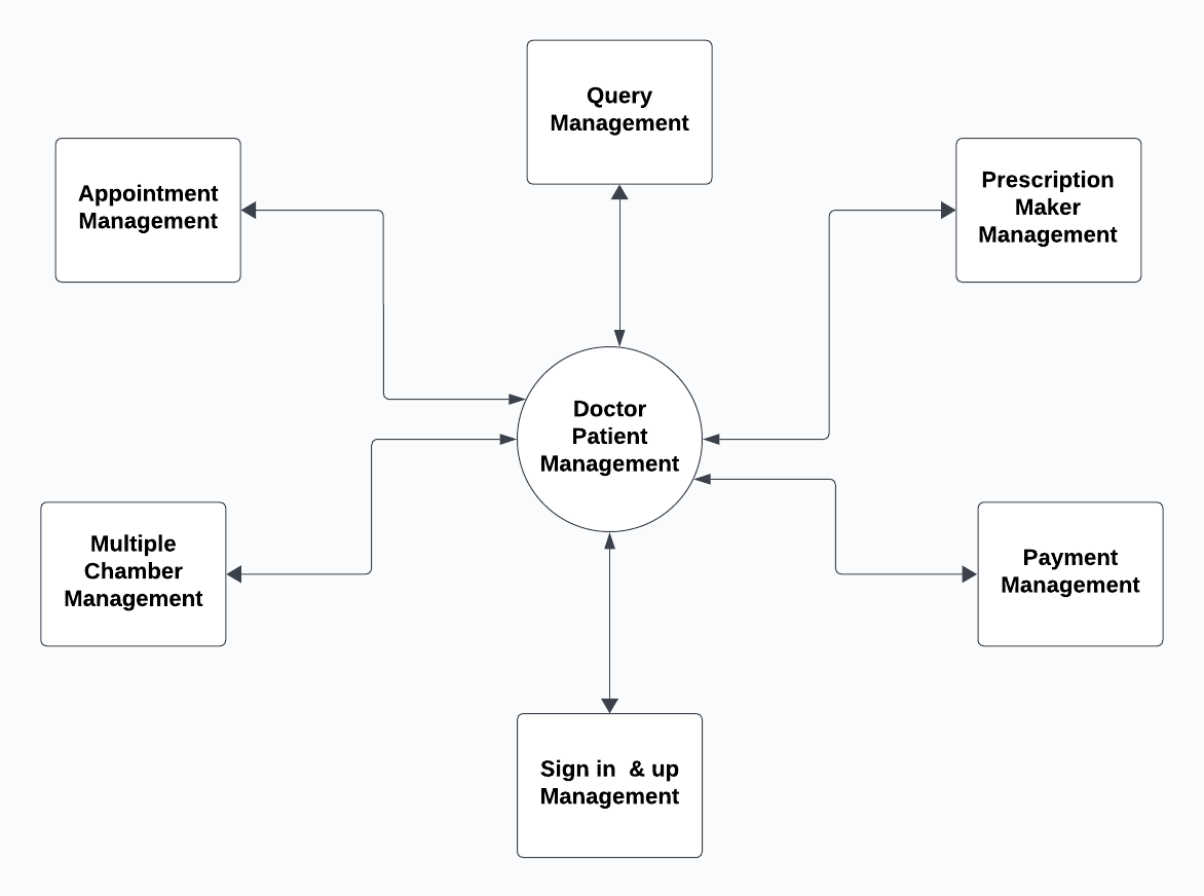

Figure 37: Zero Level Dfd For Systems

7.7.2 Data flow diagram for user

The central entity is "User," who interacts with several processes: "Log in," "Verify Email," "Forget Email," and "Modules."

1. "Log in" processes user credentials and grants access to modules upon successful verification.
2. "Verify Email" validates user emails, confirming their validity.
3. "Forget Email" handles password resets, sending instructions to verified emails.
4. "Modules" encompasses a collection of sub-processes: "Appointment," "Payment," "Profile," and "Check Previous Record."

Within "Modules": "Appointment" manages booking and scheduling appointments. "Payment" processes financial transactions. "Profile" enables users to view and update personal information. "Check Previous Record" allows users to access their historical data.

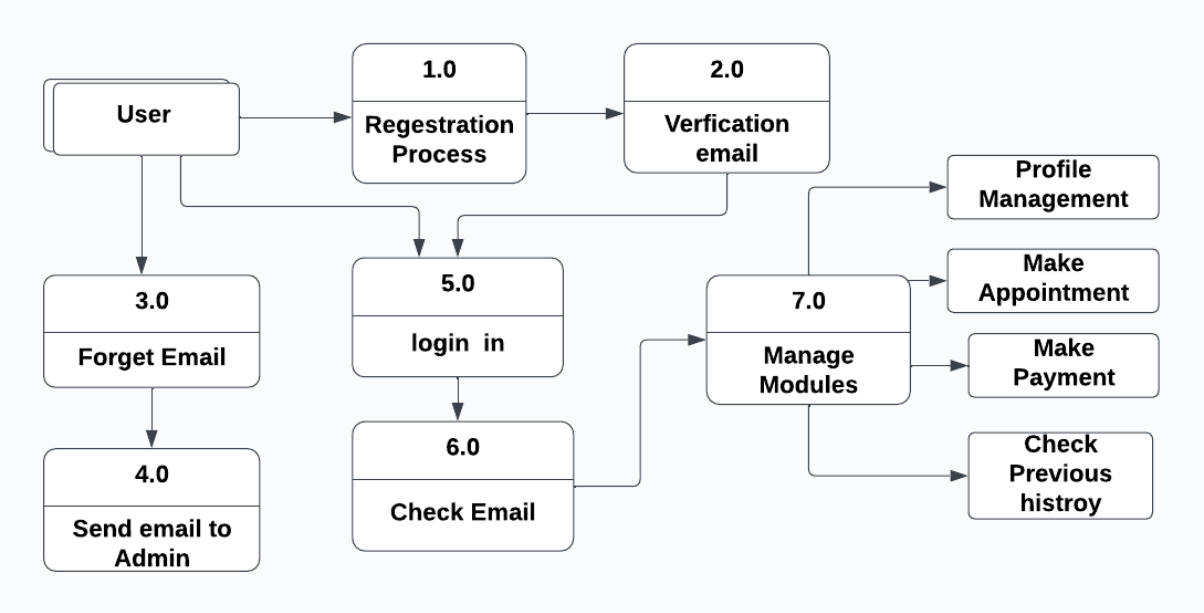

Figure 38: Data flow diagram for user

7.7.3 Data flow diagram for Admin

Admin: The primary user with elevated permissions and access to administrative functionalities. Forget Email: Assists admins with password resets, sending instructions to verified email addresses. Log in: Authenticates admin credentials, granting access upon successful verification. Roles and Access: Manages user roles and permissions, defining their access levels within the system. Modules: Encompasses a collection of sub-processes specifically designed for administrative tasks.

Modules and Their Entities:

Prescription: Manages medical prescriptions, potentially including creation, editing, and review. Profile Edit: Enables admins to modify user profiles, ensuring accurate and up-to-date information. Appointment: Oversees appointment scheduling and management, potentially including booking, rescheduling, and cancellations. Check Previous Story: Grants admins access to historical records, enabling review of past user interactions and events.

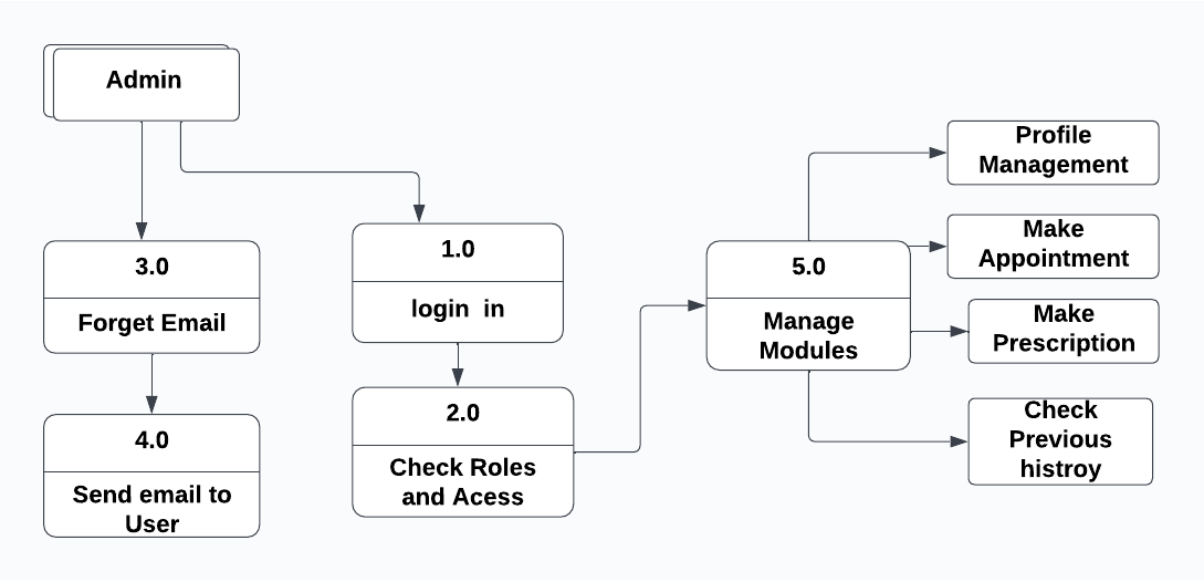

Figure 39: Data flow diagram for Admin

7.8 ER diagram

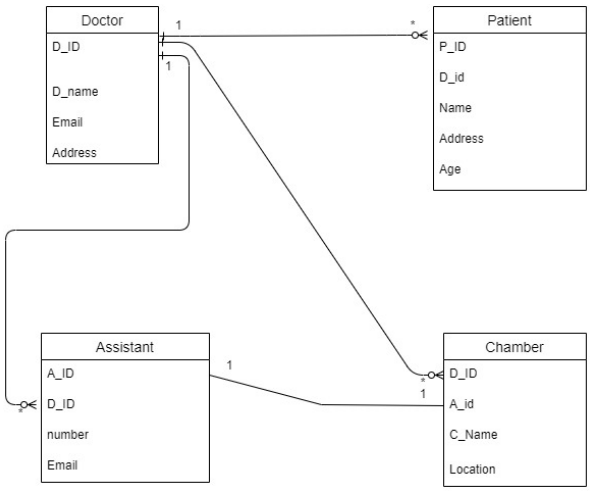

Figure 40: ER Diagram

The ER diagram shows the relationship between doctors, patients, and assistants in a medical setting.
The main entities are:
Doctor: This entity has attributes for the doctor's ID, name, email address, and address.
Patient: This entity has attributes for the patient's ID, name, address, and age.
Assistant: This entity has attributes for the assistant's ID, doctor's ID (as assistants are associated with specific doctors), phone number, and email address.
The relationships between the entities are:
Doctor can have one or many Patients. This is indicated by the line connecting the "Doctor" entity to the "Patient" entity, with the crow's foot notation on the "Patient" side. This means that a doctor can see many patients, but a patient can only see one doctor at a time.
Assistant is associated with one Doctor. This is indicated by the line connecting the "Assistant" entity to the "Doctor" entity, with the number "1" next to the line on the "Doctor" side. This means that an Doctor can work with multiple doctor.

7.9 API Documentation

1. Introduction

- The Doctor Appointment and Prescription System API allows managing information about doctors, chambers, appointments, and prescriptions. The system supports doctors with multiple chambers, allowing for effective appointment scheduling and prescription management.

2. Authentication

- All endpoints require authentication using API key-based authentication.

3. Endpoints
3.1 Register

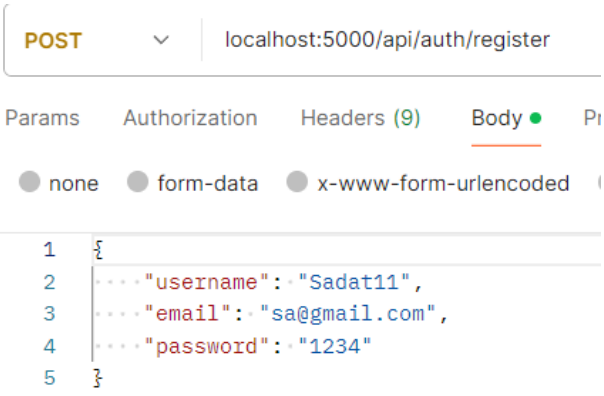

Figure 41: Register

3.2 Login

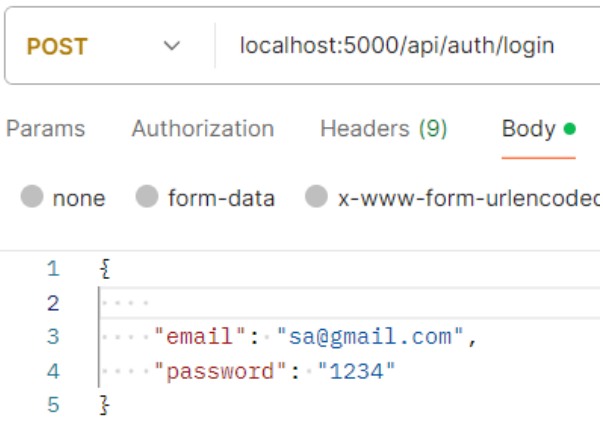

Figure 42: apilogin

3.3 Appointment Management

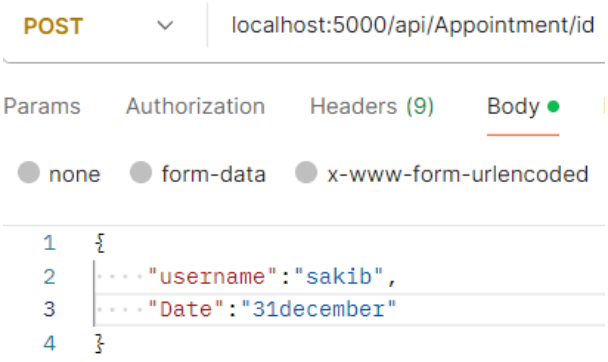

Figure 43: appointment

3.4 Payment

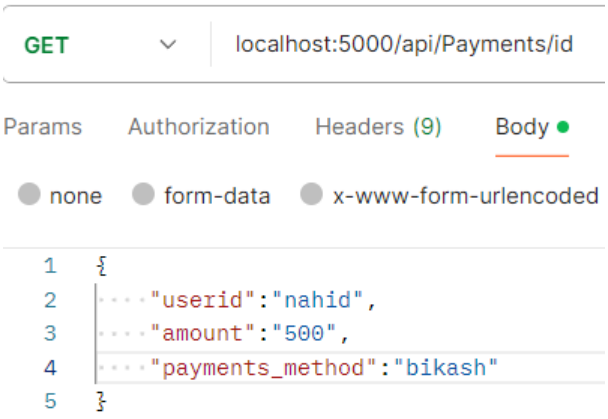

Figure 44: api payment get

6. Tools and Technology

- Authentication: API key-based authentication.
- Data Format: JSON.
- Communication Protocol: HTTP/HTTPS.
- Documentation: Swagger/OpenAPI.

7. Implementation Guidelines

- Implement secure communication using HTTPS.
- Use proper error handling and provide meaningful error messages.
- Regularly update API documentation.

7.10 component level design for the systems

Component-level design, also known as component-based software engineering (CBSE), is a software development approach that focuses on building systems using reusable components. It emphasizes dividing the system into independent, well-defined modules that interact with each other through clearly

defined interfaces. The prescription maker system boasts a user-friendly User Interface (UI) tailored for healthcare professionals, prioritizing an intuitive experience in inputting patient details, selecting medications, and generating comprehensive prescriptions. Seamlessly integrated is the Patient Information Module, proficiently managing the storage and retrieval of patient data to empower healthcare providers with swift access and updates during prescription creation. Augmenting precision, the Medication Database component stores exhaustive information on diverse medications, ensuring accurate and current selections throughout the prescription-making process. The core Prescription Generation Algorithm interprets medical inputs, guaranteeing proper dosage, medication interactions, and adherence to medical guidelines. Security is paramount with a robust module safeguarding patient and prescription data through authentication and authorization measures. Integration with Electronic Health Records (EHR) fosters collaboration, sharing prescription data for seamless continuity of patient care. Additional features include versatile Print and Export Functionality, an Audit Trail for transparency and accountability, a Notification System for critical updates, and a Reporting and Analytics Module empowering administrators with insights for continuous enhancement and informed decision-making. A component-level design for the doctor system is given below:

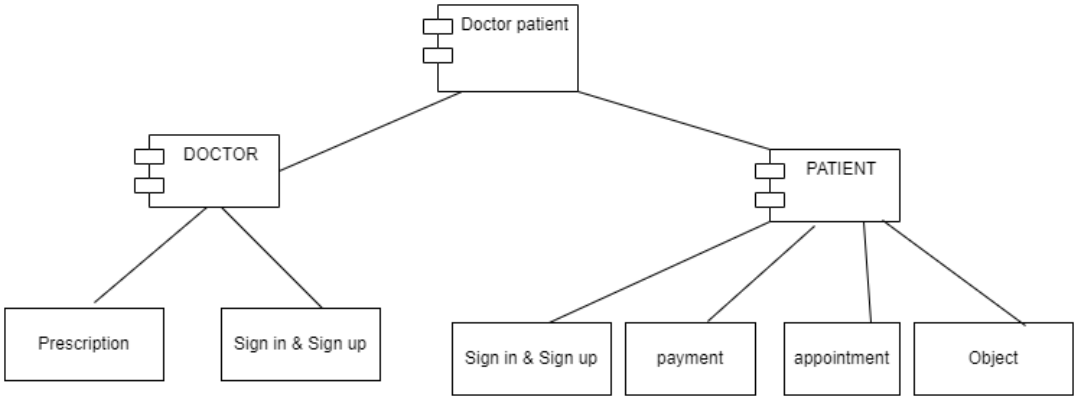

Figure 45: Component Level Design for Doctor

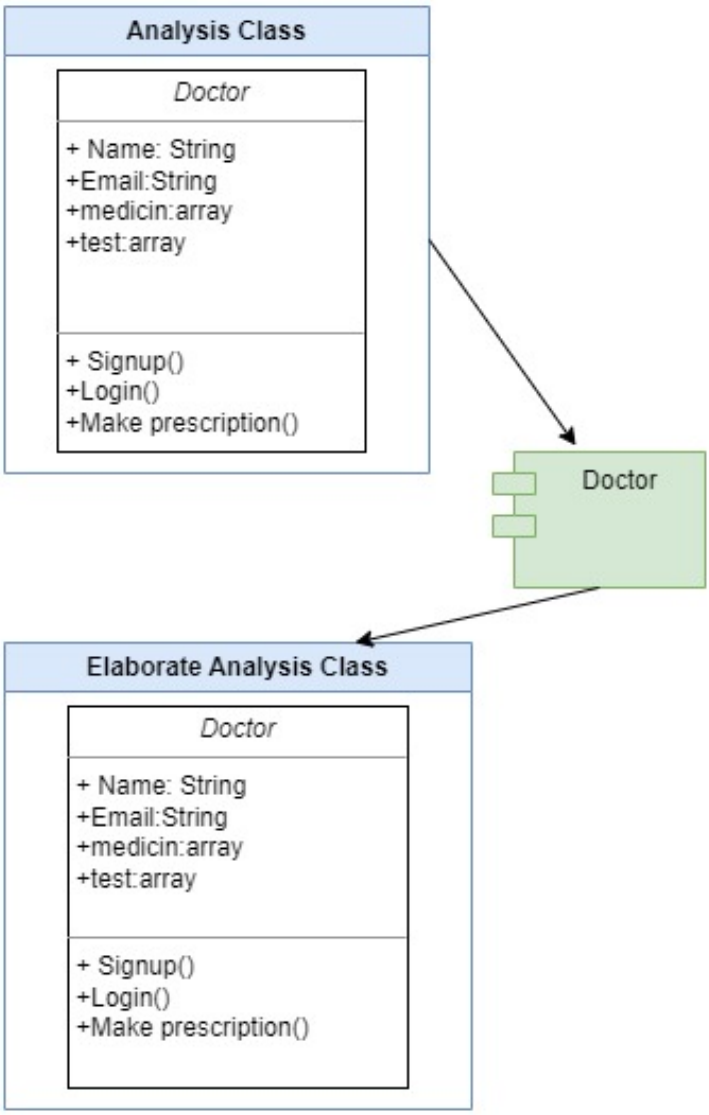

Figure 46: Component Level Design for Doctor

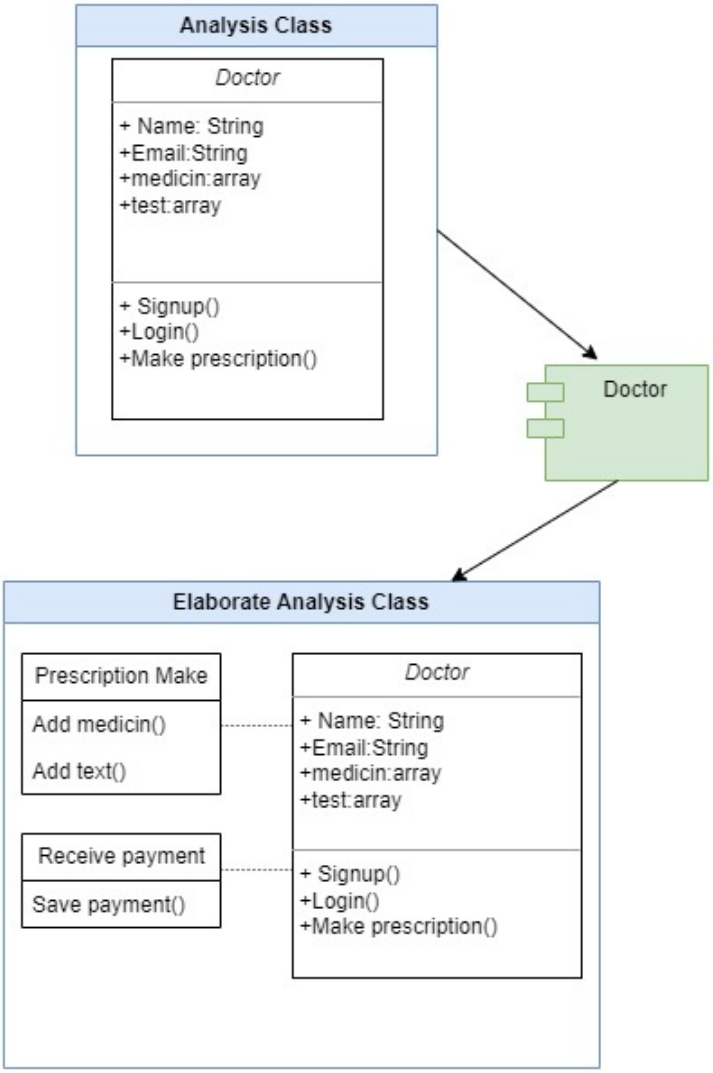

Figure 47: Component Level Design for Doctor

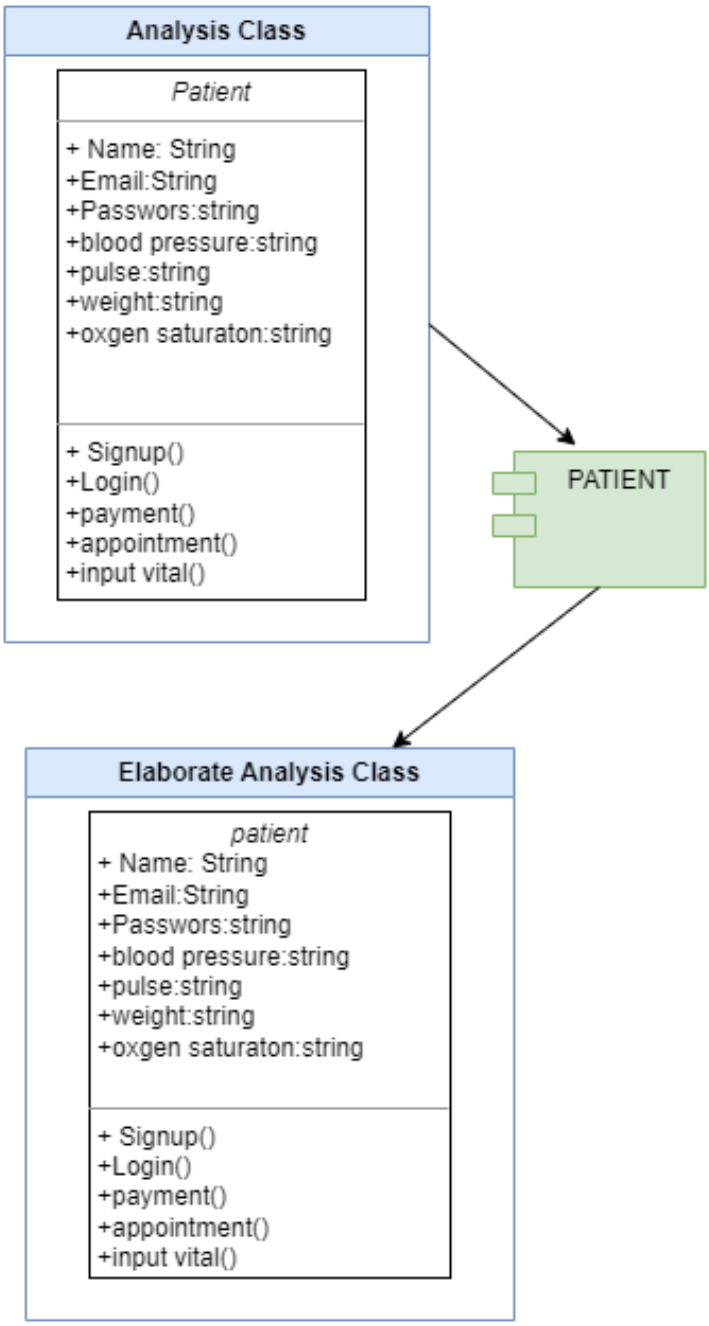

Figure 48: Component Level Design for Doctor

7.11 Test cases and Test case automation

7.11.1 Log In

- Assign a unique Test Case ID using a clear naming convention for easy reference.
- Write a concise Test Case Description that articulates the specific functionality being tested, such as verifying successful login with valid credentials.
- Define any Preconditions that must be met before executing the test case, such as having a valid user account.
- List the Test Steps in detail, outlining the actions to be performed in sequential order, such as entering username, password, and clicking the login button.
- Gather appropriate Test Data to use as input during test execution, such as valid and invalid usernames and passwords.
- Specify the Expected Result of the test case if executed successfully, such as a successful login and redirection to the user's homepage.
- Define the Post Condition that should be achieved upon successful test completion, such as the user being logged in and viewing their account dashboard.
- Record the Actual Result observed during test execution, comparing it to the expected result to determine the test case's status (Pass/Fail).

Project Name	Doctor and patient Management
Project Module	Log in
Created By	Nahid Imtiaz
Reviewed By	Nahid & Souhardyo(peers)
Date	03-01-24

Figure 49: Test cases and Test case automation for log in

7.11.2 Appointment

- Assign a unique Test Case ID using a clear naming convention for easy reference.
- Write a concise Test Case Description that specifies the exact appointment functionality being tested, such as booking a new appointment, rescheduling, or canceling.
- Define any Preconditions that must be met, such as having a valid user account or availability of appointment slots.
- List the Test Steps in detail, outlining the actions to be performed in order, such as selecting a date, time, provider, entering patient details, and confirming the appointment.
- Gather appropriate Test Data to use as input, such as valid and invalid appointment details, time slots, and patient information.
- Specify the Expected Result of each test case, such as a successful appointment booking, a confirmation message, or appropriate error handling for invalid inputs.

Test Id	Test senario	Test Case	Pre condition	Test Steps	Test Data	Expected Result	Actual Result
Tc_log in 01	Verify login of gmail	Enter valid gmail and valid password	Need valid email and valid password to do login	1.Enter Email 2.Enter Password 3.Click login button	<valid gmail> <valid password>	Succesfully login	
Tc_log in 02	Verify login of gmail	Enter invalid gmail and valid password	Need valid email and valid password to do login	1.Enter Email 2.Enter Password 3.Click login button	<invalid gmail> <valid password>	A messge from email	
Tc_log in 03	Verify login of gmail	Enter valid gmail and invalid password	Need valid email and valid password to do login	1.Enter Email 2.Enter Password 3.Click login button	<valid gmail> <invalid password>	A messge from email	
Tc_log in 04	Verify login of gmail	Enter invalid gmail and invalid password	Need valid email and valid password to do login	1.Enter Email 2.Enter Password 3.Click login button	<invalid gmail> <invalid password>	A messge from email	

Figure 50: Test cases and Test case automation for log in

- Define the Post Condition that should be achieved, such as the appointment being correctly reflected in the calendar, notifications being sent, or availability being updated.
- Record the Actual Result observed during test execution, comparing it to the expected result to determine the test case's status (Pass/Fail).

Project Name	Doctor and patient Management
Project Module	Appointment
Created By	Nahid Imtiaz
Reviewed By	Nahid & Souhardyo(peers)
Date	03-01-24

Figure 51: Test cases and Test case automation for Appointment

7.11.3 Payment

- Assign a unique Test Case ID using a clear naming convention for easy identification and tracking.
- Write a concise Test Case Description that clearly articulates the specific payment functionality being tested, such as successful payment processing, handling invalid payment details, or validating payment gateways.
- Define any Preconditions that must be met before test execution, such as having items in a cart, a valid user account with payment information, or specific account balances.

Test Id	Test senario	Test Case	Pre condition	Test Steps	Test Data	Expected Result	Actual Result
Tc_log in 01	Verify login of gmail	make payment and make date fix	Make payment for appointment	1.Make payment 2.Make date fix	<make payment> <make date fix>	Succesfully Appointment	
Tc_log in 02	Verify login of gmail	do not make payment and make date fix	Make payment for appointment	1.Make payment 2.Make date fix	<do not make payment> <make date fix>	A messge from email	

Figure 52: Test cases and Test case automation for Appointment

- List the Test Steps in detail, outlining the actions to be performed in sequential order, including entering payment details (card number, expiry date, CVV), selecting payment methods, and confirming payment.
- Gather appropriate Test Data to use as input during test execution, covering valid and invalid payment information, different payment methods (credit cards, debit cards, digital wallets), and various transaction amounts.
- Specify the Expected Result of each test case, such as successful payment confirmation, error messages for invalid inputs, accurate transaction records, and appropriate communication with payment gateways.
- Define the Post Condition that should be achieved upon successful test completion, such as updated account balances, order status changes, generation of payment receipts, and initiation of fulfillment processes.
- Record the Actual Result observed during test execution, comparing it to the expected result to determine the test case's status (Pass/Fail) and identify any discrepancies or issues.

Project Name	Doctor and patient Management
Project Module	Payment
Created By	Nahid Imtiaz
Reviewed By	Nahid & Sadat(peers)
Date	03-01-24

Figure 53: Test cases and Test case automation for Payment

7.11.4 Prescription Maker

- Assign a unique Test Case ID using a clear naming convention for easy identification and tracking.
- Write a concise Test Case Description that specifies the exact prescription functionality being tested, such as creating a new prescription, editing existing ones, managing medication details, or generating printouts.

Test Id	Test senario	Test Case	Pre condition	Test Steps	Test Data	Expected Result	Actual Result
Tc_log in 01	Verify login of gmail	login for payment	login for payment	1.Enter login 2.Enter payment	<Enter login> <.Enter payment>	Succesfully Payment	
Tc_log in 02	Verify login of gmail	do not login	login for payment	1.Enter login 2.Enter payment	<do not login > <Enter payment>	A messge from email	

Figure 54: Test cases and Test case automation for payment

- Define any Preconditions that must be met, such as having patient information loaded, authorized access to create prescriptions, or required medication data available.
- List the Test Steps in detail, outlining the actions to be performed in sequential order: Selecting a patient. Entering medication details (name, dosage, frequency, instructions). Choosing the quantity and refills. Adding any notes or warnings. Saving or printing the prescription.
- Gather appropriate Test Data to use as input, such as valid and invalid medication names, dosage ranges, patient information, and prescription formats.
- Specify the Expected Result of each test case, such as successful prescription creation, accurate medication details, proper formatting, and adherence to medical guidelines.
- Define the Post Condition that should be achieved, such as the prescription being saved correctly in the patient's record, available for viewing and printing, and potentially sent to pharmacies.
- Record the Actual Result observed during test execution, comparing it to the expected result to determine the test case's status (Pass/Fail) and identify any discrepancies or issues.

Project Name	Doctor and patient Management
Project Module	Prescribtion Maker
Created By	Nahid Imtiaz
Reviewed By	Nahid & Sadat(peers)
Date	03-01-24

Figure 55: Test cases and Test case automation for prescription maker

Test Id	Test senario	Test Case	Pre condition	Test Steps	Test Data	Expected Result	Actual Result
Tc_log in 01	Verify login of gmail	make payment and make appointment	Make payment , appointment for prescription	1.Make payment 2.Make appointment	<make payment> <make appointment>	Succesfully Prescription	
Tc_log in 02	Verify login of gmail	do not make payment and make appointment	Make payment , appointment for prescription	1.Make payment 2.Make appointment	<do not make payment> <make appointment>	A messge from email	

Figure 56: Test cases and Test case automation for prescription maker

8 Process Model Used

The Spiral model combines the idea of iterative development with the systematic aspects of the waterfall model. It is suitable for projects with high risk and uncertainty. The Spiral model includes risk assessment in each phase, allowing for mitigation strategies to be incorporated, making it a good choice for complex and large-scale projects.

Risk Management:

- Rationale: The Spiral Model explicitly incorporates risk analysis and management throughout the development process. Each cycle in the spiral represents a phase of the project, and at the beginning of each cycle, a risk analysis is performed. This allows the development team to identify and address potential risks early, reducing the likelihood of project failure.

Complex Projects:

- Rationale: The Spiral Model is well-suited for large, complex projects where uncertainties and complexities are high. Its iterative nature allows the development team to handle evolving requirements, incorporate feedback, and address unforeseen challenges gradually over multiple iterations.

Flexibility and Adaptability:

- Rationale: The Spiral Model is adaptable to changes in requirements, making it suitable for projects with evolving or unclear requirements. The iterative nature allows for adjustments to be made at each cycle based on lessons learned from the previous iterations

Customer Collaboration:

- Rationale: The model encourages customer involvement and feedback at various stages of development. This ensures that the final product aligns more closely with customer expectations, and adjustments can be made based on their input throughout the project's life cycle.

Verification and Validation:

- Rationale: Each cycle in the Spiral Model includes both development and validation phases. This ensures that verification and validation activities are integrated into the process, resulting in a more robust and reliable final product.

9 Risk Analysis

Cost Risk

Risk: Unforeseen expenses, budget overruns, or inaccurate cost estimates.

Mitigation Strategies:

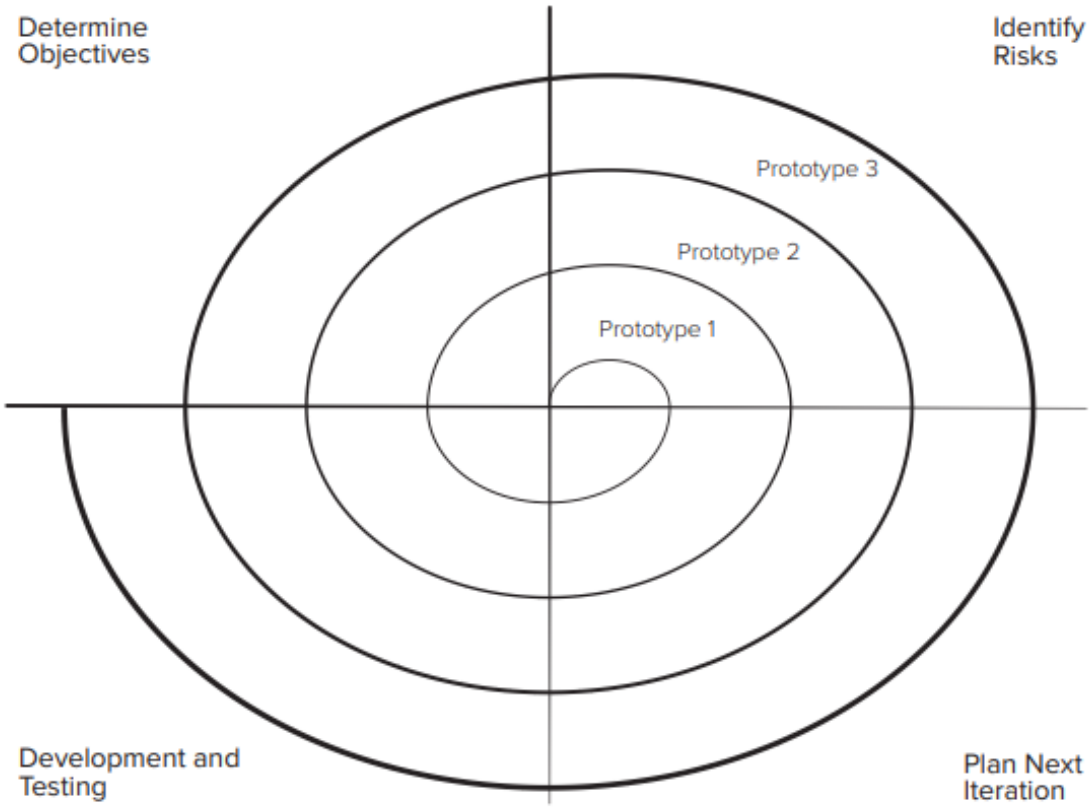

Figure 57: Process Model

- Conduct a thorough initial cost estimation with input from relevant stakeholders.
- Implement a contingency budget for unexpected expenses.
- Regularly monitor and update the budget throughout the project.

Schedule Risk

Risk: Delays in project timelines, missed milestones, or unforeseen obstacles affecting the schedule.
Mitigation Strategies:

- Develop a realistic and detailed project schedule.
- Identify critical path activities and allocate resources appropriately.
- Regularly review and update the project schedule.
- Have contingency plans in place for potential delays.

Performance Risk

Risk: Inability to meet performance requirements or deliver a product that does not meet user expectations.
Mitigation Strategies:

- Clearly define and document performance requirements.
- Conduct thorough testing at each iteration to identify and address performance issues early.

- Regularly engage with stakeholders for feedback on the product's performance.
- Implement a robust quality assurance process.

Risks	Category	Probability	Impact	RMMM
Size estimate may be significantly low	PS	60%	2	
Larger number of users than planned	PS	30%	3	
Less reuse than planned	PS	70%	2	
End users resist system	BU	40%	3	
Delivery deadline will be tightened	BU	50%	2	
Funding will be lost	CU	40%	1	
Customer will change requirements	PS	80%	2	
Technology will not meet exceptions	TR	30%	1	
Lack of training on tools	DE	80%	3	
Staff inexperienced	ST	30%	2	
Staff turnover will be high	ST	60%	2	

Figure 58: Risk Table

Support Risk

Risk: Insufficient post-implementation support, leading to challenges in maintaining and updating the system.

Mitigation Strategies:

- Develop comprehensive documentation for the system.
- Provide training to support staff and end-users.
- Establish a dedicated support team or contract with a support service.
- Implement a feedback mechanism for users to report issues or request assistance.

10 Constraints to Project Implementation

Schedule Constraints:

- The Spiral Model's iterative nature allows for flexible scheduling, accommodating changes and adjustments as required.

Budget Constraints:

- Prototyping within each spiral iteration provides opportunities for cost-effective adjustments based on real-time feedback.

Software Constraints:

- Incremental prototyping addresses compatibility issues incrementally, reducing the impact on the overall development process.

Hardware Constraints:

- Spiral Model's phased approach facilitates the gradual scaling of server infrastructure based on evolving needs.

11 Hardware and Software Resource

11.1 Hardware Resources:

Server Infrastructure:

- Determine the type and capacity of servers needed based on the project's scale.
- Consider factors such as processing power, memory, storage, and network bandwidth.

Networking Equipment:

- Plan for network infrastructure, including routers, switches, and firewalls.
- Ensure sufficient bandwidth for data transfer and communication between servers and client devices.

Storage Systems:

- Evaluate storage requirements for databases, file storage, and backups.
- Consider the type of storage (e.g., SSDs, HDDs) based on performance needs.

Backup and Recovery Systems:

- Implement backup solutions to ensure data integrity and recovery options in case of failures.
- Determine backup frequency and retention policies.

11.2 Language and Tools:

- Programming languages: Express.js and Node.js for back-end functionalities. React for frontend functionalities.
- Database: MongoDb chosen for its scalability, accommodating the evolving data storage requirements.
- Tools: Git for version control and Docker for containerization, facilitating the integration of prototypes seamlessly.

12 Project Timeline and Schedule

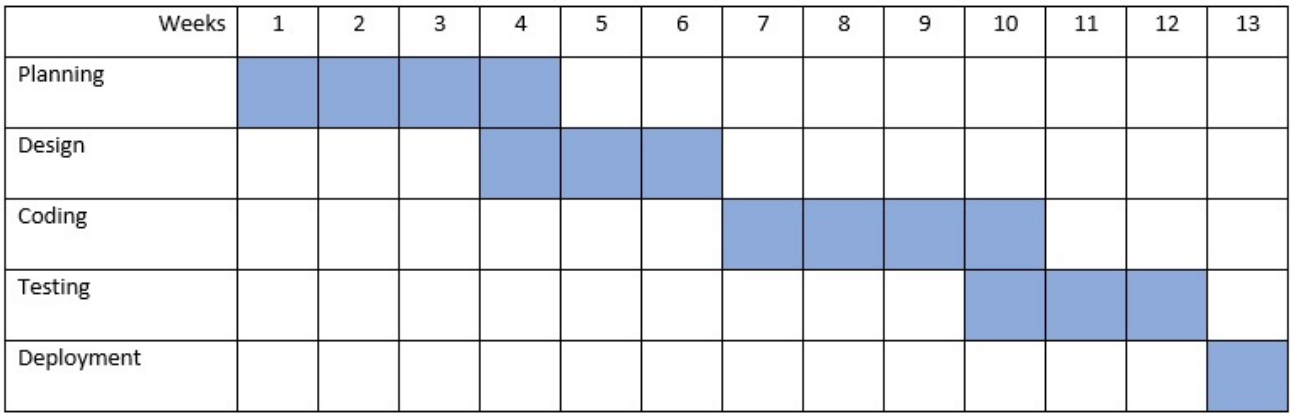

Figure 59: Project Timeline and Schedule

- Planning: 1-4 weeks.
- Design: 4-6 weeks.
- Coding: 7-10 weeks.
- Code testing: 10-12 weeks.
- Deployment: 13 week

13 Estimated Budget

There are several main factors for estimating budgets:

1. **Type of Software Project:** The web application falls under the category of a medium-sized software project.
2. **Size of Software Project:** The project size is medium, as it involves the development of A doctor patient chamber management with well-defined functionalities, addressing user needs.
   - **Small:** Minor changes to the user interface or bug fixes.
   - **Medium:** More substantial, well-defined scope, often standalone solutions or integrations.
   - **Large:** Involves depth and complexity, may require integration with multiple systems, database components, and address security and logging features.
   - **Enterprise:** Large project on steroids, exclusively built upon an underlying framework.

Area	**Specification**	**Pricing**
Resource Cost	Laptop X 3 Mobile Other Server Server	2,30,000 /= 20,000/= 4000/= 1,500/=
Developer Cost	UI/UX Designer Developer X 3	4000/= 30,000/=
Total		2,89,500/=

Figure 60: Estimation of Cost

14 Social/Cultural/Environmental impact of the project

14.1 Social Impact

Accessibility and Inclusivity:

- Positive Impact: If the project focuses on accessibility, it can positively impact people with disabilities, making technology more inclusive.
- Negative Impact: Lack of consideration for accessibility can exclude certain groups, reinforcing societal disparities.

Job Creation and Skills Development:

- Positive Impact: Successful implementation may create job opportunities and contribute to skill development in the local community.
- Negative Impact: Automation or outsourcing could lead to job displacement, impacting local employment.

Privacy and Data Security:

- Positive Impact: Robust privacy measures can enhance trust among users, protecting their sensitive information.
- Negative Impact: Data breaches or privacy concerns can erode trust and have social implications.

14.2 Cultural Impact

Cultural Sensitivity:

- Positive Impact: Consideration of cultural nuances in design and content can make the project more acceptable to diverse user groups.
- Negative Impact: Ignoring cultural sensitivities may lead to misunderstandings or resistance to the project.

Language and Localization:

- Positive Impact: Providing content in local languages and adapting to cultural norms can enhance user engagement.
- Negative Impact: Neglecting language diversity may limit accessibility and alienate non-English speakers.

14.3 Environmental impact

Sustainability Practices:

- Positive Impact: Implementing sustainable practices, such as energy-efficient technologies, can reduce the project's environmental footprint.
- Negative Impact: Ignoring sustainability may contribute to increased energy consumption and environmental degradation.

Carbon Footprint:

- Positive Impact: Employing eco-friendly technologies and practices can reduce the project's carbon footprint.
- Negative Impact: High energy consumption or resource-intensive processes may contribute to environmental pollution.

15 Impact on Individuals and Society

15.1 Impact on Individuals

Enhanced Patient Experience:

- Positive Impact: The project seeks to improve the overall experience for individuals seeking healthcare services. Patients can benefit from streamlined appointment scheduling, reduced waiting times, and improved communication with healthcare providers.

Access to Timely Medical Care:

- Positive Impact: The system aims to reduce barriers to healthcare access by enabling individuals to schedule appointments conveniently. This can result in timely medical care, early detection of health issues, and better overall health outcomes.

Empowerment through Information:

- Positive Impact: The project includes features for patients to access their medical records, test results, and prescriptions. Empowering individuals with easy access to their healthcare information can lead to better-informed decisions about their health.

15.2 Impact on Society

Efficient Healthcare Administration:

- Positive Impact: The project contributes to the efficiency of healthcare administration by automating processes such as appointment scheduling, prescription management, and medical record keeping. This can lead to more effective use of resources and improved overall healthcare system performance.

Reduced Healthcare Costs:

- Positive Impact: By streamlining processes and reducing administrative overhead, the project can contribute to the reduction of overall healthcare costs. This is beneficial for both individuals and society, making healthcare more affordable and accessible.

Data-Driven Healthcare Planning:

- Positive Impact: The system collects and analyzes healthcare data, providing valuable insights for healthcare planning and resource allocation. This can contribute to more informed decision-making at the societal level and lead to improved public health outcomes.