Software Design Description for

Noor Hatem Hassan , Nourhan Amr , Shaden Essam , Jomana Ayman, Malak Wael
Supervised by: Dr Esraa Abd el-Aziz , TA Nada Ayman , TA Ahmed Talaat

December 23, 2024

Table 1: Document version history

Version	**Date**	**Reason for Change**
1.0	23-12-2024	SDD first version's description are defined.

GitHub: https://github.com/noorharidy19/HealthCareSystem.git

Contents

1 Introduction **4**

1.1 Purpose . . . 4
1.2 Scope . . . 4
1.3 Overview . . . 4
1.4 Intended audience . . . 4
1.5 Reference Material . . . 5

2 System Overview **5**

2.1 System Scope . . . 7
2.2 System objectives . . . 8
2.3 System Timeline . . . 8

3 Design viewpoints **9**

3.1 Context viewpoint . . . 9
3.2 Tabular Description for use case . . . 12
3.3 Logical viewpoint . . . 16
3.4 Patterns use viewpoint . . . 28
3.5 Algorithm viewpoint . . . 29
3.6 Activity Diagram viewpoint . . . 29
3.6.1 User Login Activity Diagram . . . 29
3.6.2 Admin User Management Activity Diagram . . . 30
3.6.3 Admin Appointments Management Activity Diagram . . . 31
3.7 Interaction viewpoint . . . 35
3.7.1 Patient Sequence Diagram . . . 35
3.7.2 Admin Sequence Diagram . . . 38
3.7.3 Doctor Sequence Diagram . . . 39
3.8 Interface viewpoint . . . 40

4 Data Design **40**

4.1 Data Description . . . 40
4.2 Dataset Description . . . 40
4.3 Database design description . . . 41

5 Human Interface Design **41**

5.1 User Interface . . . 41
5.1.1 Patient . . . 41
5.1.2 Doctor . . . 42
5.1.3 Admin . . . 42
5.2 Screen Images . . . 42
5.3 Screen Objects and Actions . . . 45

6 Requirements Matrix **45**

7 APPENDICES **46**
7.1 Github 47
7.2 Other appendices as appropriate 48

Abstract

The healthcare chatbot system aims to improve patient engagement and health management through an intuitive platform. It allows patients to schedule doctor appointments easily Additionally, the system provides clear explanations of lab results and can access a support group chat via chatbot and medications, enabling patients to better understand their health information. By combining these features, the healthcare chatbot supports patients in managing their health proactively while enhancing communication with healthcare providers. This comprehensive tool seeks to empower users to take charge of their health and well-being.

1 Introduction

1.1 Purpose

This document focuses on the Software Design Description(SDD) for the HealthCare Chatbot System. It provides detailed information of the system architecture, including data design, diagrams Also, it includes requirement matrix which link system functionalities to their requirements. It serves as a blueprint for implementing functions such as chatbot interactions, patient appointments scheduling, medical diagnosis, explaining lab results and providing support groups for patients.

1.2 Scope

This SDD defines the comprehensive design viewpoints and system architecture for HealthCare Chatbot System. The scope includes translating the functional and non functional requirements into requirement metric , detailed description of system structure , interactions and behavior through diagrams such as use case , sequence and activity diagram. This document provides design details for the website's interactive , explanation and support groups supporting clear implementation and development

1.3 Overview

This SDD document is divided into 6 sections , each addressing different aspects of the project. To start with, Introduction section provides an overview of the project purpose , scope, overview, intended audience , reference material , definitions and acronyms. Next, system overview section provides an outline of system structure, system objectives and system timeline accompanied by a visual diagram. Next step, Design viewpoints it provides various diagrams context diagram in context viewpoint , class diagram and explanations of each class in logical viewpoint , Design pattern in patterns use viewpoint , sequence diagrams in Interaction viewpoint . In forth section, the design and description of dataset and database are discussed. Moreover, the human interface design is elaborated in fifth section. Finally , the requirement matrix is presented in sixth section outlining the system's functional requirements

1.4 Intended audience

This document is intended for the following audiences:

Software Developers:

They will use this document to implement the system according to the design. **System Architects:**

They ensure the system design aligns with the overall infrastructure and performance needs. **Project Managers:**

They will review the design to ensure the project stays on track and meets requirements. **Quality Assurance (QA) Engineers:**

They will use the design to create test plans and verify the system's functionality. **Database Administrators (DBAs):**

They will manage the database and ensure data integrity based on the design. **Healthcare Professionals (End Users):**

They may refer to this document to understand how the system fits into their workflows. **System Maintenance and Support Teams:**

They will use this document to maintain and troubleshoot the system after launch. **Regulatory Compliance Auditors:**

They ensure the system meets regulatory and legal standards.

1.5 Reference Material

Software Requirement Specification Document for Health Care System

2 System Overview

Our healthcare chatbot system provides a comprehensive solution for managing patient health by offering appointment scheduling, explaining lab results , providing support groups , accessing medical records and quick medical access.

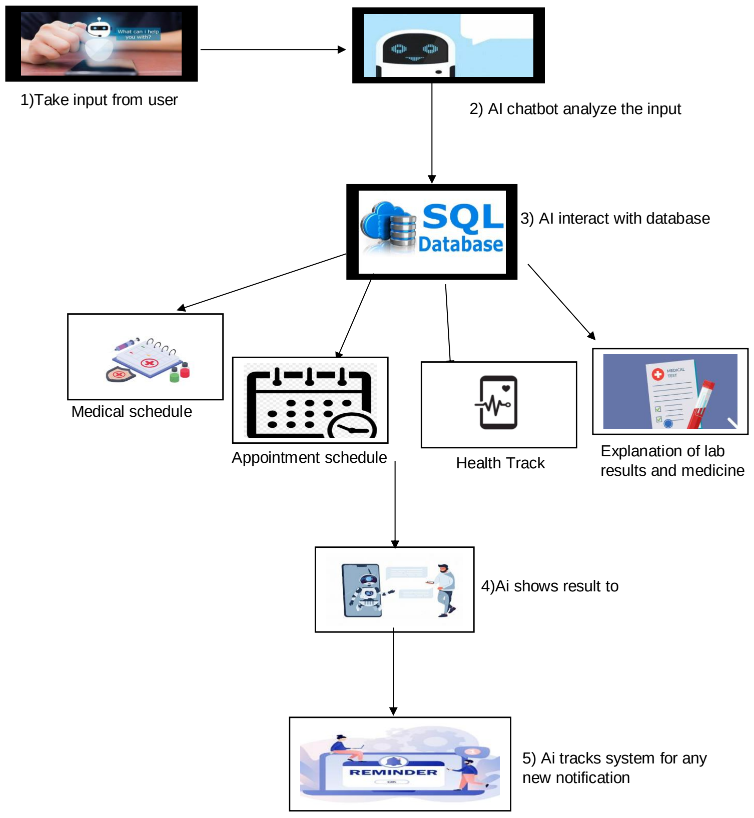

1)Take input from user
2) AI chatbot analyze the input
3) AI interact with database
Medical schedule
Appointment schedule
Health Track
Explanation of lab results and medicine
4)Ai shows result to
5) Ai tracks system for any new notification

Figure 1: System Overview for our project

1. The process begins when the user interacts with the chatbot through a web interface. The user can insert his medical records, ask for an appointment,ask for explanations about their lab results or prescriptions, or ask to join support group.

2. The chatbot analyzes the user input to determine the request. The AI understands what the user wants to do, whether it's booking a doctor's appointment, enter medical records, getting medical information, or providing support groups.

3. Once the chatbot understands the user's request, it communicates with a secure database to retrieve relevant information. For instance:

- **Appointment Scheduling**: The system accesses available appointment slots and presents them to the user.
- **Health Tracking**: The database stores the user's medical records allowing chatbot to fetch and store it in database.
- **Explanation of Lab Results & Medications**: If the user needs an explanation of lab results or prescribed medications, the system explain it to provide a clear summary.
- **Join Support Group**: The system helps patients to join support groups with other patients with the same medical condition.

4. After retrieving and analyzing the necessary data, the chatbot responds to the user with the requested information.

5. The system continuously tracks medication schedules, generating alerts when any missed medication doses occur.

2.1 System Scope

The healthcare chatbot system will include the following features:

1. **Appointment Booking**: Allows patients to easily schedule, reschedule, or cancel doctor appointments through a chat interface or by filling a form.
2. **Mediacal records**: Enables patients to input their medical records.
3. **Providing support groups**: Enables patients to join support groups with other patients with the same health condition.
4. **Medical Information Assistance**: Provides easy-to-understand explanations of lab results and medications, answering patient questions through the chatbot.
5. **User-Friendly Interaction**: Creates a simple and intuitive chat experience that allows users to easily navigate and access the information they need.
6. **Doctor Access**: Allows doctors to access their appointments, cancel appointments, and view patient data.
7. **Admin Dashboard**: Includes an admin dashboard to manage users, overseeing patient and doctor accounts to ensure smooth operation of the system.

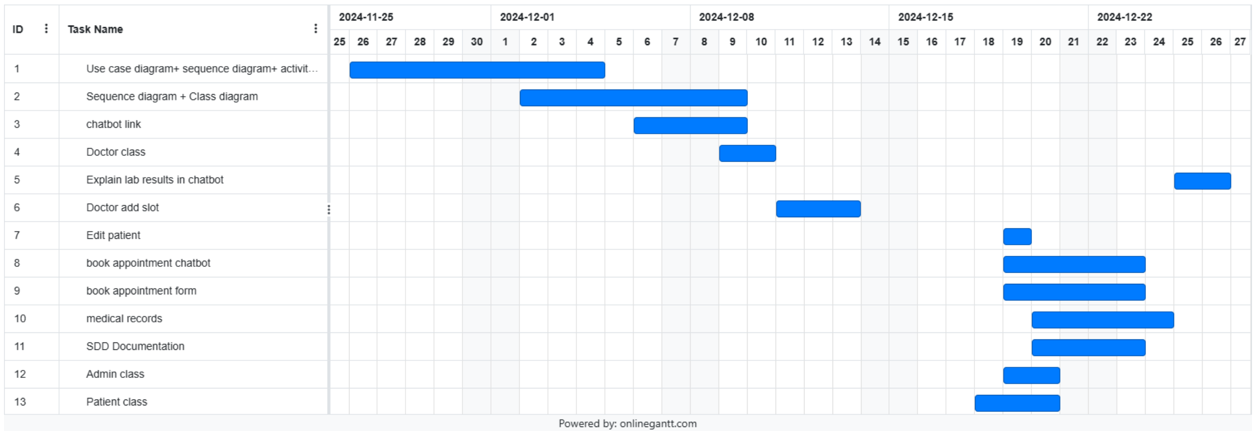
2.2 System objectives

In our project we aim to: The Healthcare Chatbot System interacts with various external entities to provide patients with real-time healthcare assistance. This system context diagram outlines the primary external actors—patients, doctors, the chatbot AI and demonstrates how they interact with the system. The system also works with a database to save patient records and retrieve information when needed. This allows the chatbot to give accurate answers. Doctors can access the system to view assigned patients,add their available time slots and manage appointments . This setup ensures that healthcare data is handled efficiently and delivered quickly to the users. Patients can input requests for services such as scheduling medication reminders, receiving explanations of lab results, tracking health metrics, and booking appointments. The system forwards these requests to the chatbot AI, which analyzes the input to determine the user's intent. The chatbot then returns relevant results to the system.

1. **Increase Patient Engagement:**
   - Make it easy for patients to get involved in managing their health with regular reminders and support.
2. **Simplify Appointment Management:**
   - Help patients easily book and keep track of doctor appointments.
3. **easy uploading Health records:**
   - Allow patients to upload their medical records to be easily accessed by doctor and patient.
4. **Educate Patients:**
   - Explain lab results and medications in simple language to help patients understand their health better through the chatbot system.
5. **Enhance Communication:**
   - Provide an easy way for patients to ask questions and communicate with healthcare providers through the chatbot system.
6. **Gather User Feedback:**
   - Collect feedback from users to improve the chatbot's features and make it more helpful.

2.3 System Timeline

This section provides the latest version of the project plan between SRS and Technical Phase, including the major tasks to be accomplished, their inter-dependencies, and their tentative start/stop dates.



ID	Task Name
1	Use case diagram+ sequence diagram+ activit...
2	Sequence diagram + Class diagram
3	chatbot link
4	Doctor class
5	Explain lab results in chatbot
6	Doctor add slot
7	Edit patient
8	book appointment chatbot
9	book appointment form
10	medical records
11	SDD Documentation
12	Admin class
13	Patient class

Figure 2: Gantt chart

3 Design viewpoints

3.1 Context viewpoint

The Healthcare Chatbot System interacts with various external entities to provide patients with real-time healthcare assistance. This system context diagram outlines the primary external actors—patients, doctors, the chatbot A—and demonstrates how they interact with the system. The system also works with a database to save patient records and retrieve information when needed. This allows the chatbot to give accurate answers. Doctors can access the system to view patient data and manage appointments an add heir available time slots. This setup ensures that healthcare data is handled efficiently and delivered quickly to the users. Patients can input requests for services such as scheduling medication reminders, receiving explanations of lab results, and booking appointments. The system forwards these requests to the chatbot AI, which analyzes the input to determine the user's intent. The chatbot then returns relevant results to the system.

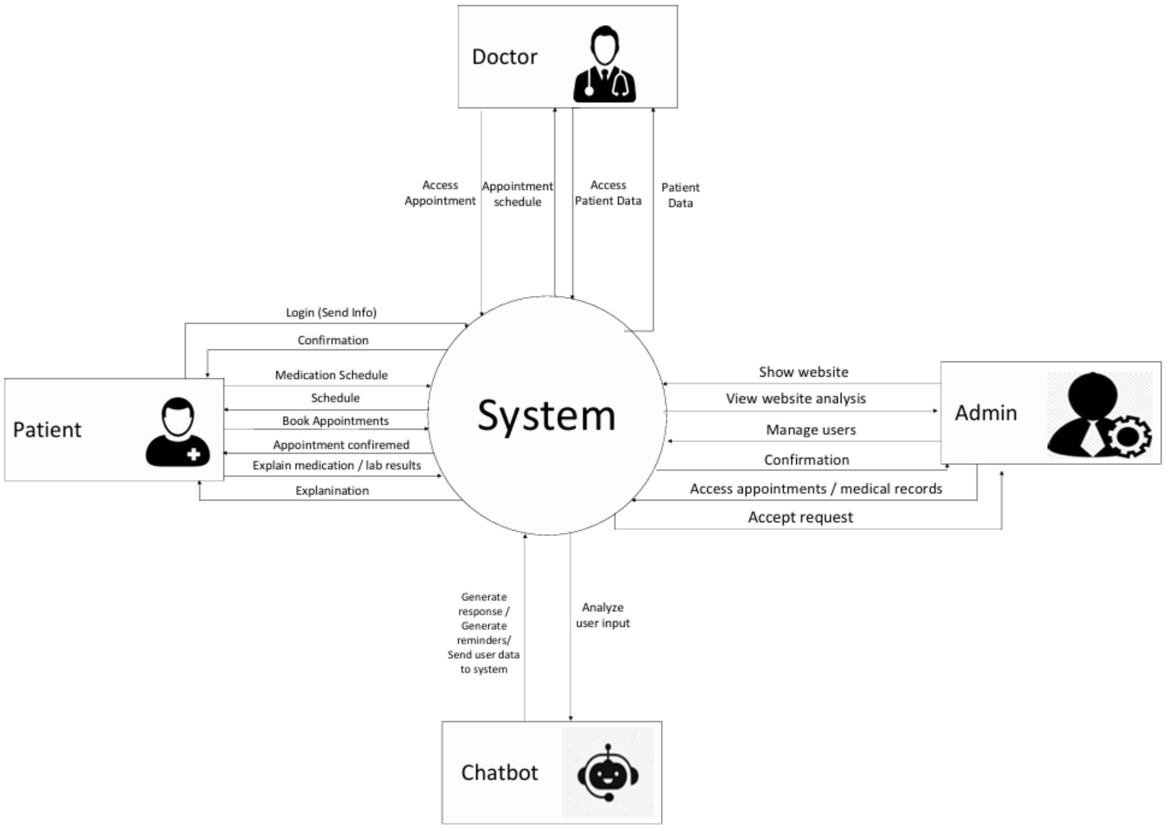


Figure 3: Context Diagram for healthcare system

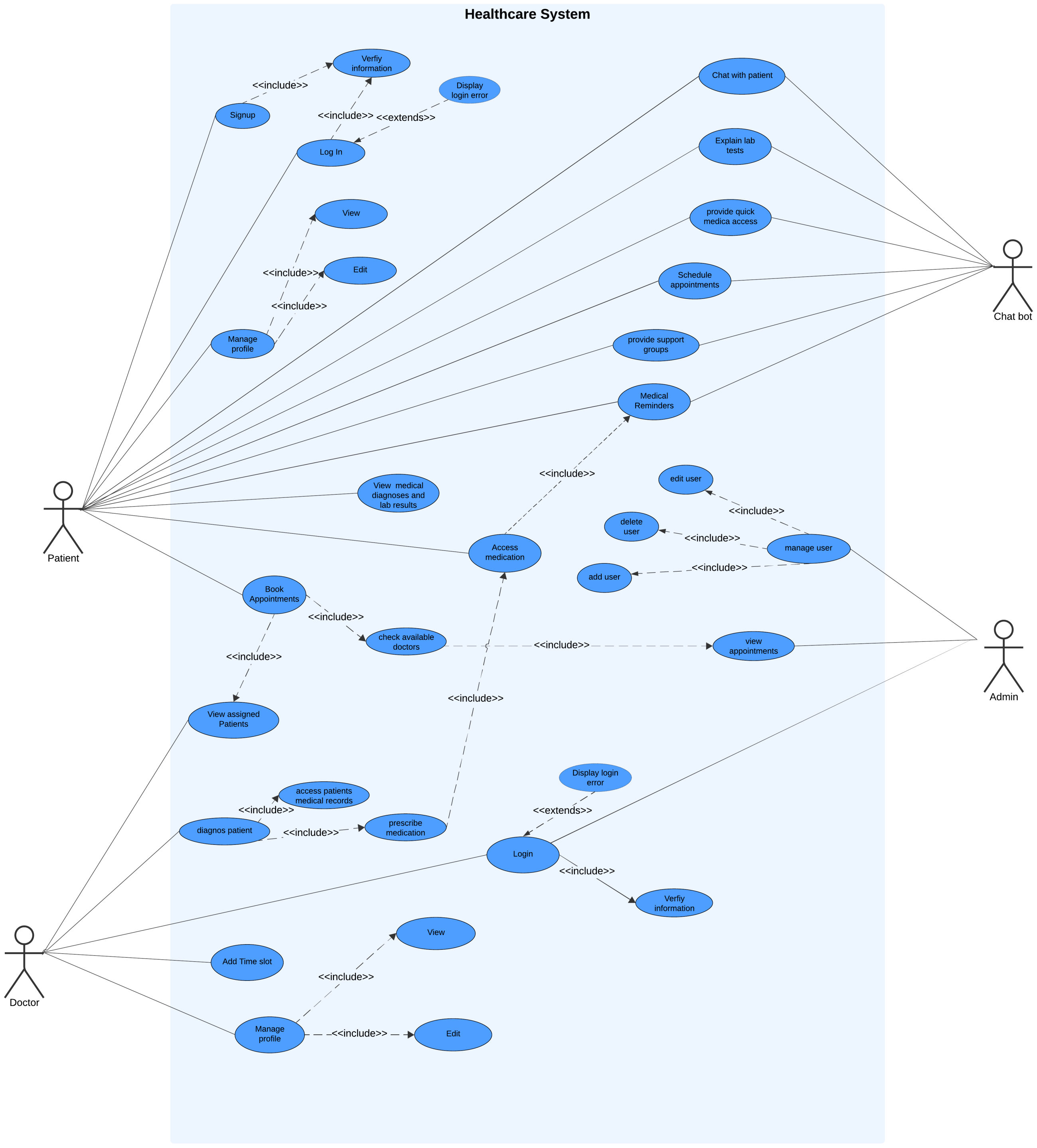


Figure 4: Use Case Diagram Example

3.2 Tabular Description for use case

Sign Up

Actor	Patient
Description	Allow new patients to sign up for our healthcare system.
Data	Patient personal information.
Stimulus	Patient clicks on the register form.
Response	Patient created successfully.
Comments	Enter valid information.

Login

Actor(s)	Patient, Admin, Doctor
Description	Allow users to access the system.
Data	User email and password.
Stimulus	User clicks on the login form.
Response	Logged in successfully.
Comments	Enter valid email and password.

Manage Profile

Actor(s)	Patient, Doctor
Description	Allow users to view, edit, or delete their personal information.
Data	Personal information.
Stimulus	User must be logged in and click on the profile option.
Response	Profile updated or deleted successfully.
Comments	User must have a valid account and be logged in.

Book Appointments

Actor	Patient
Description	Patients can book an appointment and the system shows available appointment slots.
Data	Patient medical records and condition.
Stimulus	Click on "Book Appointments."
Response	Appointment booked successfully.
Comments	User must enter their medical records.

Access Medication

Actor	Patient
Description	Patients can access their prescribed medication.
Data	Patient prescription.
Stimulus	Click on "Access Medication."
Response	Medication details are displayed.
Comments	Patient must have booked an appointment and been prescribed medication.

View Medical Diagnosis and Lab Results

Actor	Patient
Description	Patients can access their medical diagnosis and lab results.
Data	Diagnosis and lab results provided by the doctor.
Stimulus	Click on "View Medical Diagnosis and Lab Results."
Response	Diagnosis and lab results are displayed.
Comments	Patient must have completed lab tests and been diagnosed by a doctor.

Chat with Patients

Actor(s)	Patient, Chatbot
Description	Patients can interact with the chatbot for healthcare-related queries.
Data	Patient’s query or message.
Stimulus	Click on the chatbot icon.
Response	Chatbot provides relevant responses based on the patient’s request.
Comments	Patients must be logged in to access the chatbot.

Provide Quick Medical Access

Actor(s)	Patient, Chatbot
Description	The chatbot provides instant responses to medical queries.
Data	Patient’s query or message.
Stimulus	Click on the chatbot icon.
Response	Chatbot provides the most relevant response.
Comments	Patients must be logged in to access the chatbot.

Provide Support Group

Actor(s)	Patient, Chatbot
Description	Patients can communicate with others who share similar medical conditions.
Data	Patient messages and group information.
Stimulus	Click on "Join Support Group."
Response	Access to the support group is granted.
Comments	Patients should have similar medical conditions.

Schedule Appointments

Actor(s)	Patient, Chatbot
Description	Patients can book appointments through the chatbot.
Data	Appointment details.
Stimulus	Patient interacts with the chatbot and selects the "Book Appointment" option.
Response	Appointment is successfully scheduled.
Comments	Chatbot must check for available appointment slots.

Explain Lab Tests

Actor(s)	Patient, Chatbot
Description	Chatbot explains lab test results in a patient-friendly way.
Data	Lab test results provided by the patient.
Stimulus	Click on the chatbot icon.
Response	Chatbot explains the lab results.
Comments	Patients must be logged in to access the chatbot.

View Assigned Patients (Doctor)

Actor	Doctor
Description	Doctors can view their assigned patients and appointments.
Data	Patient information and appointment details.
Stimulus	Click on "View My Patients."
Response	Displays a list of assigned patients.
Comments	Doctors must be logged in and have assigned patients.

Diagnose Patients

Actor	Doctor
Description	Doctors can diagnose patients based on medical records, conditions, and lab results.
Data	Patient information and lab results.
Stimulus	Click on "Diagnose This Patient."
Response	Patient is diagnosed successfully.
Comments	Doctors must be logged in and have assigned patients.

Manage Users (Admin)

Actor	Admin
Description	Admin can manage user accounts (add, update, or remove users).
Data	User information.
Stimulus	Click on "Add/Edit/Remove User."
Response	User account is successfully updated.
Comments	Admin must be logged in.

View Appointments (Admin)

Actor	Admin
Description	Admin can view scheduled appointments for both doctors and patients.
Data	Appointment details.
Stimulus	Click on "View Appointments."
Response	Displays a list of scheduled appointments.
Comments	Admin must be logged in, and appointments must already exist.

.

3.3 Logical viewpoint

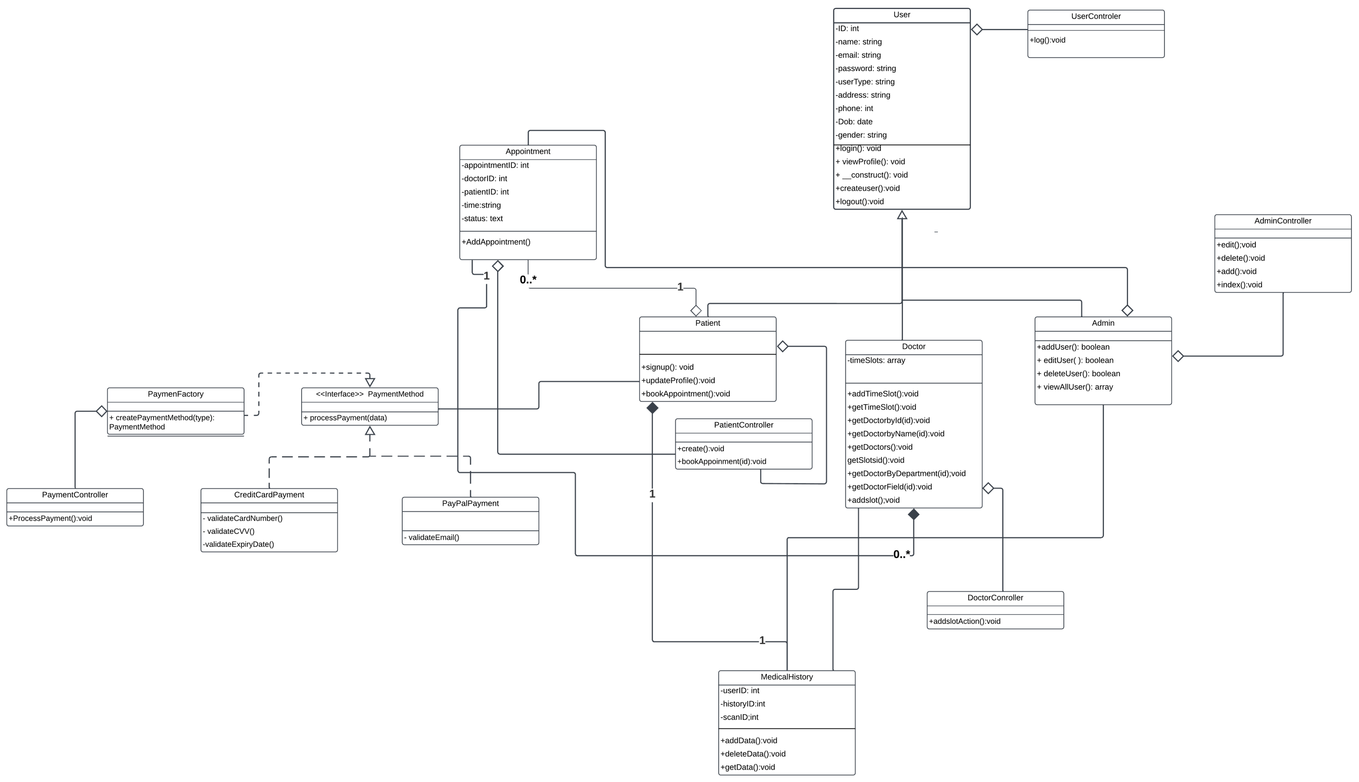


Figure 5: Class Diagram

Table 2: User class

Abstract or Concrete:	concreate
Superclasses	-
Subclasses	Patient,Doctor,Admin
Purpose	The ‘User‘ class represents a general user in the healthcare system. It serves as the base class for the ‘Patient‘, ‘Doctor‘, and ‘Admin‘ subclasses, containing common attributes and operations for all users, such as personal details, login credentials, and user type.
Collaborations	The ‘User‘ class collaborates with the subclasses
Attributes	ID:int name:string email:string password:string userType:string address:string phone:string Dob:string gender:string
Operations	**Login()** Allow users to enter his email and password. **ViewProfile()** Allow users to access their profile **loadUserData()** prints users data **saveUser()** save user information **createUser()** creates new user **getUserInfo()** Return user information **getUserAppointments()** return user appointment **logout()** destroy users session

Table 3: Patient class

Abstract or Concrete:	concreate
Superclasses	User
Subclasses	
	Purpose The ‘Patient‘ class represents patients in the healthcare system.The class ensures that patient-related activities,such as scheduling appointments,explaining lab results, joining support group , prescibe medication and expalaining medications are handled securely and efficiently.
Collaborations	The ‘Patient‘ class collaborates with several other classes within the healthcare system: • Collaborates with the ‘Appointment‘ class to allow patients to book and manage their appointments with doctors. • Works with the ‘Doctor‘ class to manage doctor-patient relationships.
Attributes	
Operations	**Signup()** Allow users to register to our website,

Table 4: Doctor class

Abstract or Concrete:	concreate
Superclasses	User
Subclasses	
Purpose	The ‘Doctor‘ class represents doctors in the healthcare system. It extends the ‘User‘ class and manages doctor-specific operations, including managing appointments, viewing assigned patients, adding available appointment slots.
Collaborations	The ‘Doctor‘ class collaborates with several other classes within the healthcare system: • Collaborates with the ‘Appointment‘ class to manage patient appointments, allowing doctors to view, accept, or cancel appointments. • Works with the ‘Patient‘ class to access patient information.
Attributes	timeSlots:array
Operations	**addTimeSlot()** Allow doctor to add time slot for their appointments. **getTimeSlots()** retrive all avaliable slots. **getDoctorById()** get specefic doctor using id **addSlot()** add slot in database **getDoctorFields()** retrive doctor fields **getDoctorsByDepartment()** show doctors based on department choosed **getSlotDetails()** retrive slots details **getSlots()** retrive all slots **getDoctors()** retrive all doctors **getDoctorIdByName()** retrive doctor id by doctor name

Table 5: User Controller class

Abstract or Concrete:	concreate
Superclasses	
Subclasses	
Purpose	The ‘UserController‘ class is responsible for managing interactions between the user interface (UI) and the user model in the healthcare system. It handles requests related to user actions. This controller acts as a mediator between the system’s views and user data, ensuring smooth user-related operations.
Collaborations	The ‘UserController‘ class collaborates with this component of the system: • Collaborates with the ‘User‘ class to access and modify user data in the system.
Attributes	
Operations	**log()** checks wether the user is successful logged in or not.

Table 6: Appointment class

Abstract or Concrete:	concreate
Superclasses	
Subclasses	
Purpose	The ‘Appointment‘ class is responsible for managing appointments between patients and doctors within the healthcare system.
Collaborations	The ‘Appointment‘ class collaborates with other system components to facilitate appointment management: • Collaborates with the ‘Patient‘ class to link patients to their scheduled appointments. • Works with the ‘Doctor‘ class to assign doctors to appointments and update their schedules.
Attributes	AppointmentID:int patientID:int doctorID:int time:int status:string
Operations	**addAppointment()** Allow patient to add appointment.

Table 7: Admin class

Abstract or Concrete:	concreate
Superclasses	User
Subclasses	
Purpose	The ‘Admin‘ class is responsible for managing users, within the healthcare system. The Admin can add, edit, delete, and retrieve user information, ensuring that the system maintains accurate and up-to-date records of all users, including patients, doctors, and other admins.
Collaborations	The ‘Admin‘ class collaborates with other system components for comprehensive user and system management: • Collaborates with the ‘User‘ class to manage and manipulate user records in the system. • Interacts with the ‘Database‘ component to store, update, and delete user data. • Works with the ‘Report‘ class to generate system activity and user-related reports.
Attributes	
Operations	**deleteUser()** Allow admin to delete users. **editUser()** Allow admin to edit users. **getUser()** Allow admin to retrive all users. **addUser()** Allow admin to add new doctor or admin users.

Table 8: Admin Controller class

Abstract or Concrete:	concreate
Superclasses	
Subclasses	
Purpose	The ‘AdminController‘ class is responsible for managing administrative tasks in the healthcare system. It allows administrators to oversee and control critical system operations, including user management, appointment reports, doctor activity, and more. This controller helps admins perform actions such as adding, deleting, and updating user records and viewing system reports for informed decision-making.
Collaborations	The ‘AdminController‘ class collaborates with several components of the system: • Collaborates with the ‘User‘ class to manage user data, allowing the admin to add, delete, or update users. • Works with the ‘Doctor‘ and ‘Patient‘ classes to monitor and retrieve data related to doctors and patients. • Interacts with the ‘Database‘ component to retrieve reports on system usage, appointments, and other statistics.
Attributes	
Operations	**edit()** edit user by accessing database
Operations	**delete()** delete user by accessing database
Operations	**add()** add users by accessing database
Operations	**viewAllUser()** view all users by accessing database

Table 9: Doctor Controller class

Abstract or Concrete:	concreate
Superclasses	
Subclasses	
Purpose	The ‘DoctorController‘ class manages operations related to doctors in the healthcare system. It allows doctors to add available time slots for patients to book, and view the details of patients assigned to them. The controller ensures that doctors can efficiently manage their availability and patient interactions through the system.
Collaborations	The ‘DoctorController‘ class collaborates with: • The ‘Doctor‘ class to retrieve and update the doctor’s information, including their schedules and assigned patients. • The ‘Appointment‘ class to manage and track patient appointments for the doctor. • The ‘Patient‘ class to view and manage the information of patients assigned to the doctor. • The ‘View‘ component to display the doctor’s appointment schedules, available slots, and patient details.
Attributes	
Operations	**addslotaction()** adds slot to database

Table 10: Patient Controller class

Abstract or Concrete:	concreate
Superclasses	
Subclasses	
Purpose	The ‘PatientController‘ class manages operations related to patients within the healthcare system. It allows patients to book, and create new patient. The controller ensures that patients have easy access to appointment management and their medical data.
Collaborations	The ‘PatientController‘ class collaborates with: • The ‘Patient‘ class to retrieve and update the patient’s personal information and health records. • The ‘Appointment‘ class to manage the booking and cancellation of appointments. • The ‘Doctor‘ class to access information on doctors available for appointments. • The ‘View‘ component to display the patient’s appointments, personal health information, and available doctors.
Attributes	
Operations	**create()** adds new user to database. **bookappointment()** add new appointment to the database.

Table 11: Medical history class

Abstract or Concrete:	concreate
Superclasses	
Subclasses	
Purpose	The ‘MedicalHistory‘ class stores and manages the health history of a patient. It provides detailed records of a patient’s medical background to aid doctors in treatment and diagnosis.
Collaborations	The ‘MedicalHistory‘ class collaborates with: • The ‘Patient‘ class to retrieve the medical history associated with a specific patient. • The ‘Admin‘ class to edit in medical records.
Attributes	UserID:int historyID:int scanID:int
Operations	**addData()** add medical records to database **deleteData()** delete medical records to database **getData()** get all medical records from database

Table 12: Payment Method class

Abstract or Concrete:	interface
Superclasses	
Subclasses	
Purpose	The ‘Payment Method‘ class is responsible for processing payments for appointments or services in the healthcare system. It utilizes the **Factory Pattern** to allow for the easy addition of new payment methods in the future, by dynamically creating payment objects without modifying existing code.
Collaborations	The ‘Payment‘ class collaborates with: • The ‘Appointment‘ class to handle payment after booking an appointment. • The ‘Factory‘ class to instantiate specific payment types (e.g., CreditCard, PayPal) based on user input or system requirements.
Attributes	
Operations	**processpayment()** takes paymenttype and create it by factory method.

Table 13: PayPal payment class

Abstract or Concrete:	interface
Superclasses	
Subclasses	
Purpose	The ‘PayPalPayment‘ class implements the ‘Payment‘ class to process payments through PayPal. It includes the logic specific to PayPal’s payment gateway and can be used in the healthcare system to handle patient transactions.
Collaborations	• The ‘PaymentFactory‘ class to instantiate and return the ‘PayPalPayment‘ object.
Attributes	
Operations	**validateEmail()** checks wether email is correct or not.

Table 14: CreditCard payment class

Abstract or Concrete:	interface
Superclasses	
Subclasses	
Purpose	This class handles the payment process via credit card, including validating the card details and processing the payment. It simulates interaction with a payment gateway to execute the transaction.
Collaborations	This class collaborates with the Payment class to extend payment processing functionality. It interacts with external payment gateways (simulated here) to complete the transaction.
Attributes	
Operations	**vaidateCardNumber()** checks wether card number valid or not. **vaidateCVV()** checks wether CVV valid or not. **vaidateExpiryDate()** checks wether Expiry date valid or not.

Table 15: Payment Factory class

Abstract or Concrete:	interface
Superclasses	
Subclasses	
Purpose	The PaymentFactory class provides a factory method to create payment objects based on the selected payment method. It simplifies adding new payment methods (like PayPal, CreditCard) without modifying the client code.
Collaborations	The PaymentFactory class collaborates with various payment classes such as PayPalPayment and CreditCardPayment to create payment objects dynamically.
Attributes	
Operations	**createPaymentMethod()** takes the new payment object and create it.

Table 16: Payment Controller class

Abstract or Concrete:	interface
Superclasses	
Subclasses	
Purpose	The PaymentController class handles user requests related to payment processing. It uses the PaymentFactory to create payment objects and coordinates the payment process (either via PayPal or CreditCard).
Collaborations	The PaymentController collaborates with the PaymentFactory to create the payment object and interacts with various Payment methods (PayPalPayment, CreditCardPayment) to process payments.
Attributes	
Operations	**createPaymentMethod()** takes paymenttype and create it by factory method.

3.4 Patterns use viewpoint

Design patterns in Health-care system: Design patterns are essential tools for creating reusable and maintainable code. They allow developers to refactor and improve the structure of their code without altering its core logic, ensuring efficiency and scalability.

- **Adapter pattern**, which is a structural design pattern that allows objects with incompatible interfaces to collaborate. how did we use it: We combined PHP and Python files to create the chatbot's logic. An adapter was implemented to facilitate seamless communication between the two, enabling efficient interaction with the bot.

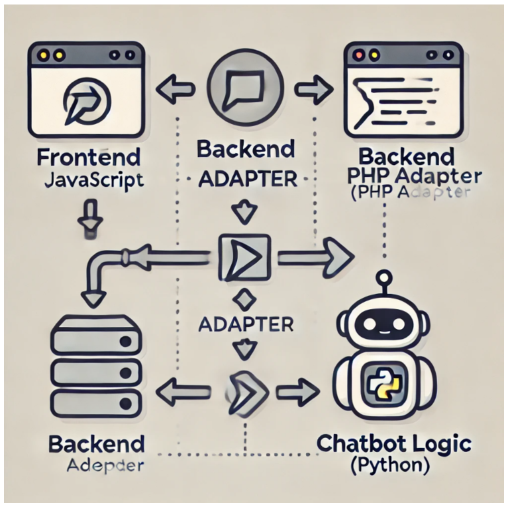


Figure 6: Adapter Pattern

Factory pattern: is a design pattern used to create objects without specifying the exact class of the object being created. Instead, it defines a method in a parent class (or interface) that is overridden by subclasses to decide which specific object to create.

- The PaymentFactory is responsible for creating payment methods based on the provided type (e.g., "CreditCard" or "PayPal").
- The client code calls PaymentFactory.createPaymentMethod(type) and receives a specific instance of PaymentMethod.
- Each concrete class (CreditCardPayment or PayPalPayment) provides its own implementation of the processPayment(data) method and any additional validations specific to the payment method.
- This pattern ensures that new payment methods can be added without altering the existing factory or client code, adhering to the Open/Closed Principle.

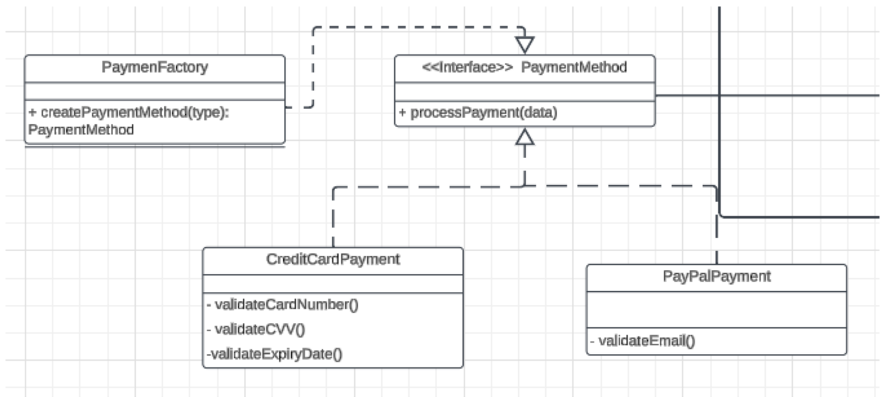


Figure 7: Factory Pattern

3.5 Algorithm viewpoint

3.6 Activity Diagram viewpoint

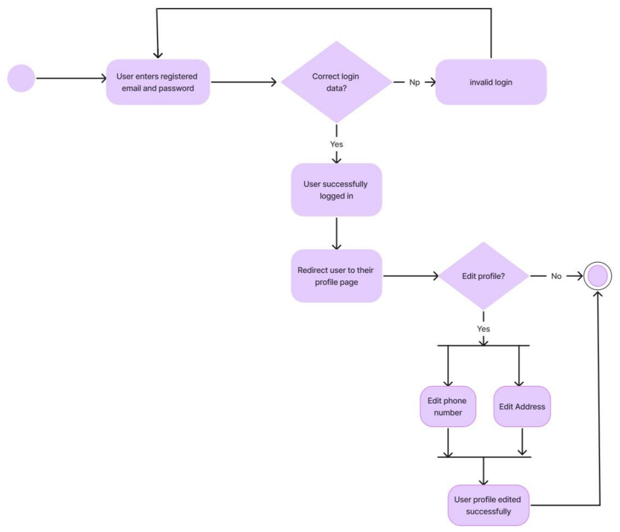

User enters registered email and password
Correct login data?
Np
invalid login
Yes
User successfully logged in
Redirect user to their profile page
Edit profile?
No
Yes
Edit phone number
Edit Address
User profile edited successfully

Figure 8: User Login Activity Diagram

3.6.1 User Login Activity Diagram

- This activity diagram represents the action flow a user takes when logging into the healthcare chatbot system, then the system checks if the login data entered is valid, if valid then user is directed to their profile page where they have the option to edit their profile information. On the other hand, if not valid an error message occurs.

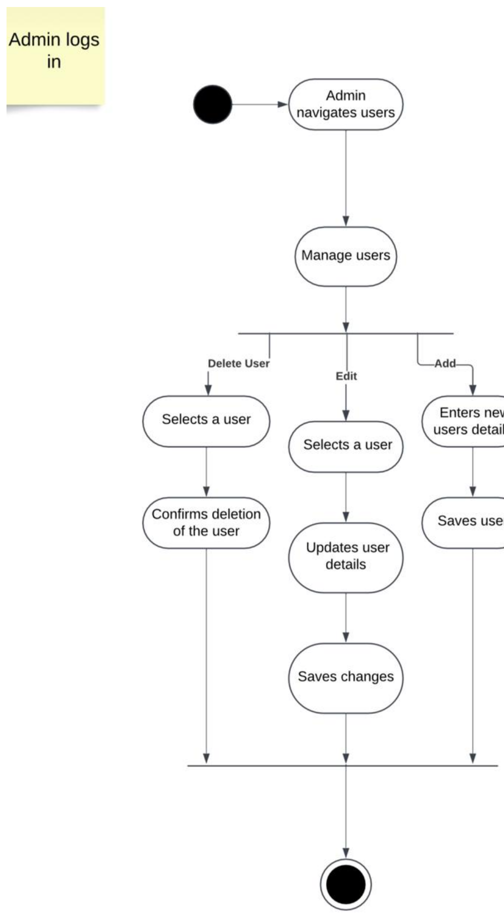


Figure 9: Admin User Management Activity Diagram

3.6.2 Admin User Management Activity Diagram

- The diagram represents the process of an admin managing users (patients,doctors) within the healthcare chatbot system. The admin has the authority to perform the following operation: delete, edit, and add a user.

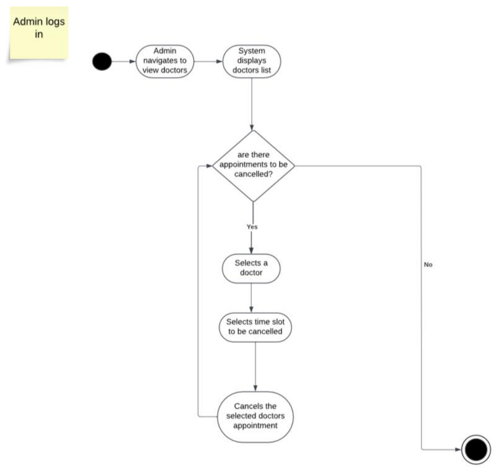

Admin logs in
Admin navigates to view doctors
System displays doctors list
are there appointments to be cancelled?
Yes
No
Selects a doctor
Selects time slot to be cancelled
Cancels the selected doctors appointment

Figure 10: Admin Appointments Management Activity Diagram

3.6.3 Admin Appointments Management Activity Diagram

- The diagram represents the admins authentication to cancel a specific doctors appointment, where the admin selects a doctor then selects the time slot to cancel which leads to canceling the selected doctor appointment.

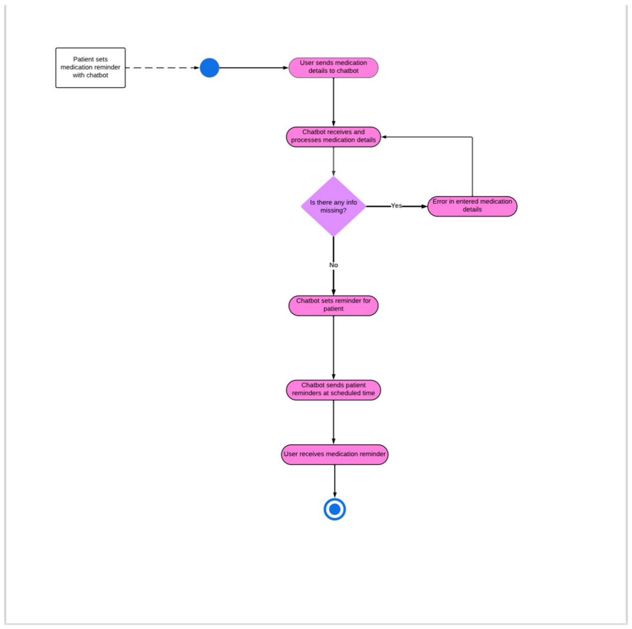

Patient sets medication reminder with chatbot
User sends medication details to chatbot
Chatbot receives and processes medication details
Is there any info missing?
Yes
Error in entered medication details
No
Chatbot sets reminder for patient
Chatbot sends patient reminders at scheduled time
User receives medication reminder

Figure 11: Medication Reminder Activity Diagram

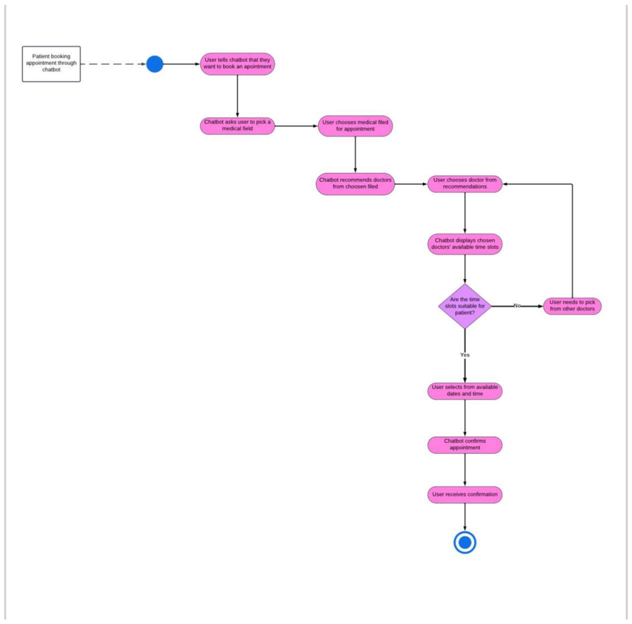

Patient booking appointment through chatbot
User tells chatbot that they want to book an apointment
Chatbot asks user to pick a medical field
User chooses medical filed for appointment
Chatbot recommends doctors from choosen filed
User chooses doctor from recommendations
Chatbot displays chosen doctors' available time slots
Are the time slots suitable for patient?
No
User needs to pick from other doctors
Yes
User selects from available dates and time
Chatbot confirms appointment
User receives confirmation

Figure 12: Appointment Booking Activity Diagram

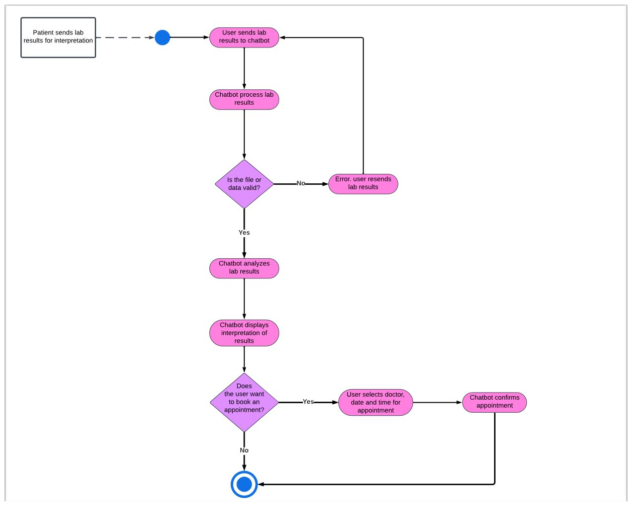

Patient sends lab results for interpretation
User sends lab results to chatbot
Chatbot process lab results
Is the file or data valid?
No
Error, user resends lab results
Yes
Chatbot analyzes lab results
Chatbot displays interpretation of results
Does the user want to book an appointment?
Yes
User selects doctor, date and time for appointment
Chatbot confirms appointment
No

Figure 13: Lab Results Activity Diagram

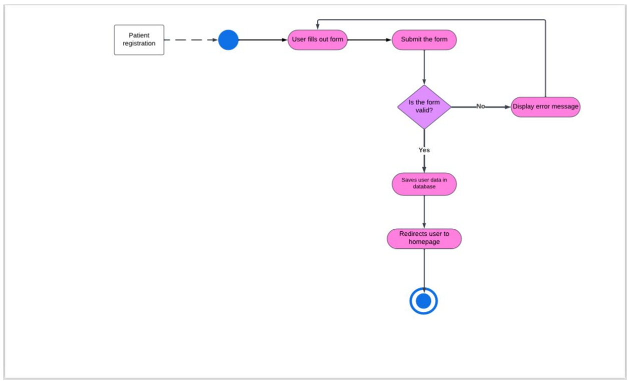


Figure 14: Patient Registration Activity Diagram

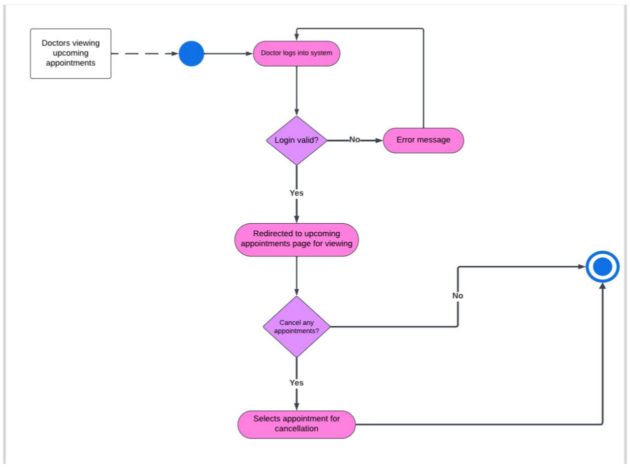

Doctors viewing upcoming appointments
Doctor logs into system
Login valid?
No
Error message
Yes
Redirected to upcoming appointments page for viewing
Cancel any appointments?
No
Yes
Selects appointment for cancellation

Figure 15: Doctor Viewing Appointments Activity Diagram

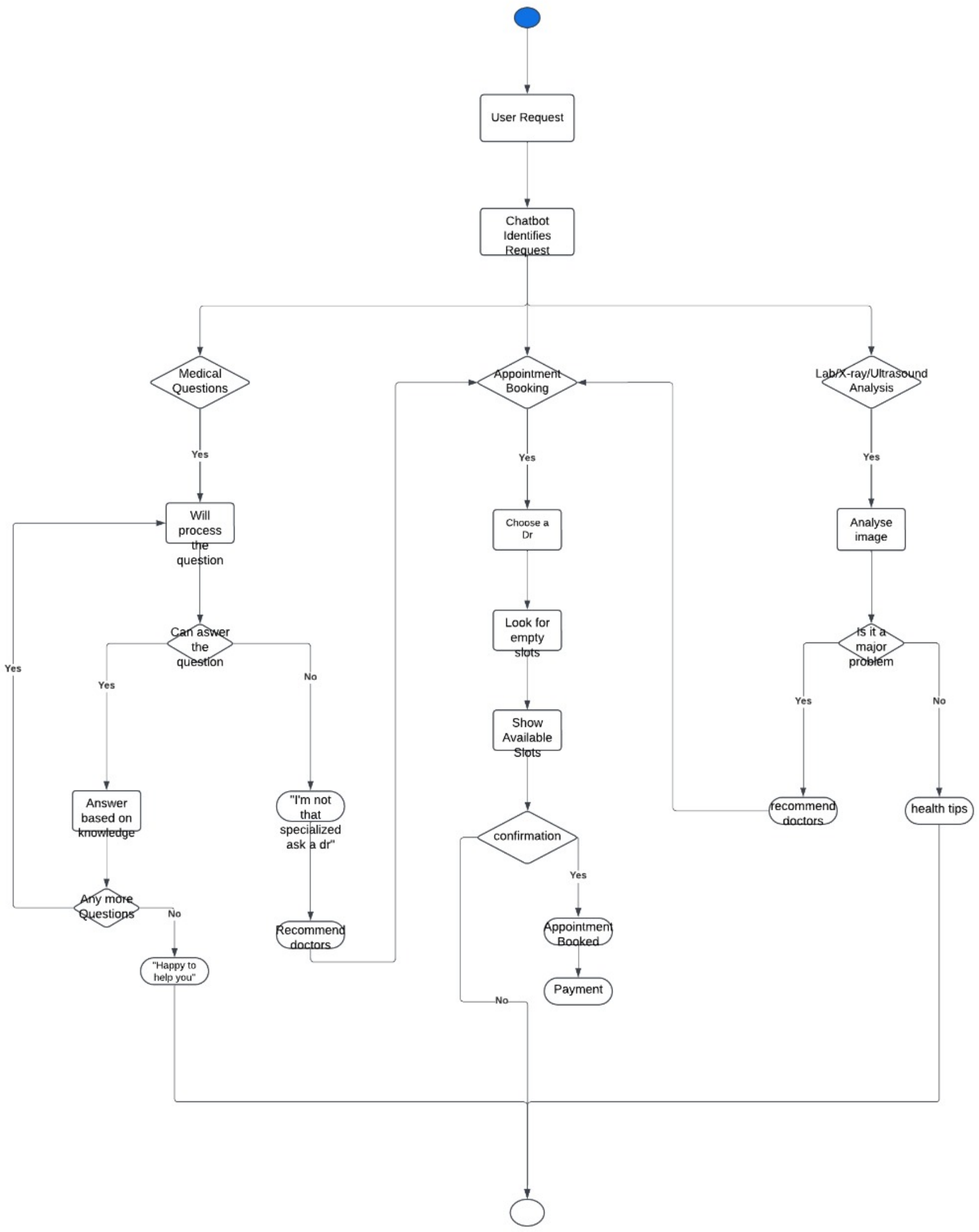


Figure 16: Chatbot activity diagram

3.7 Interaction viewpoint

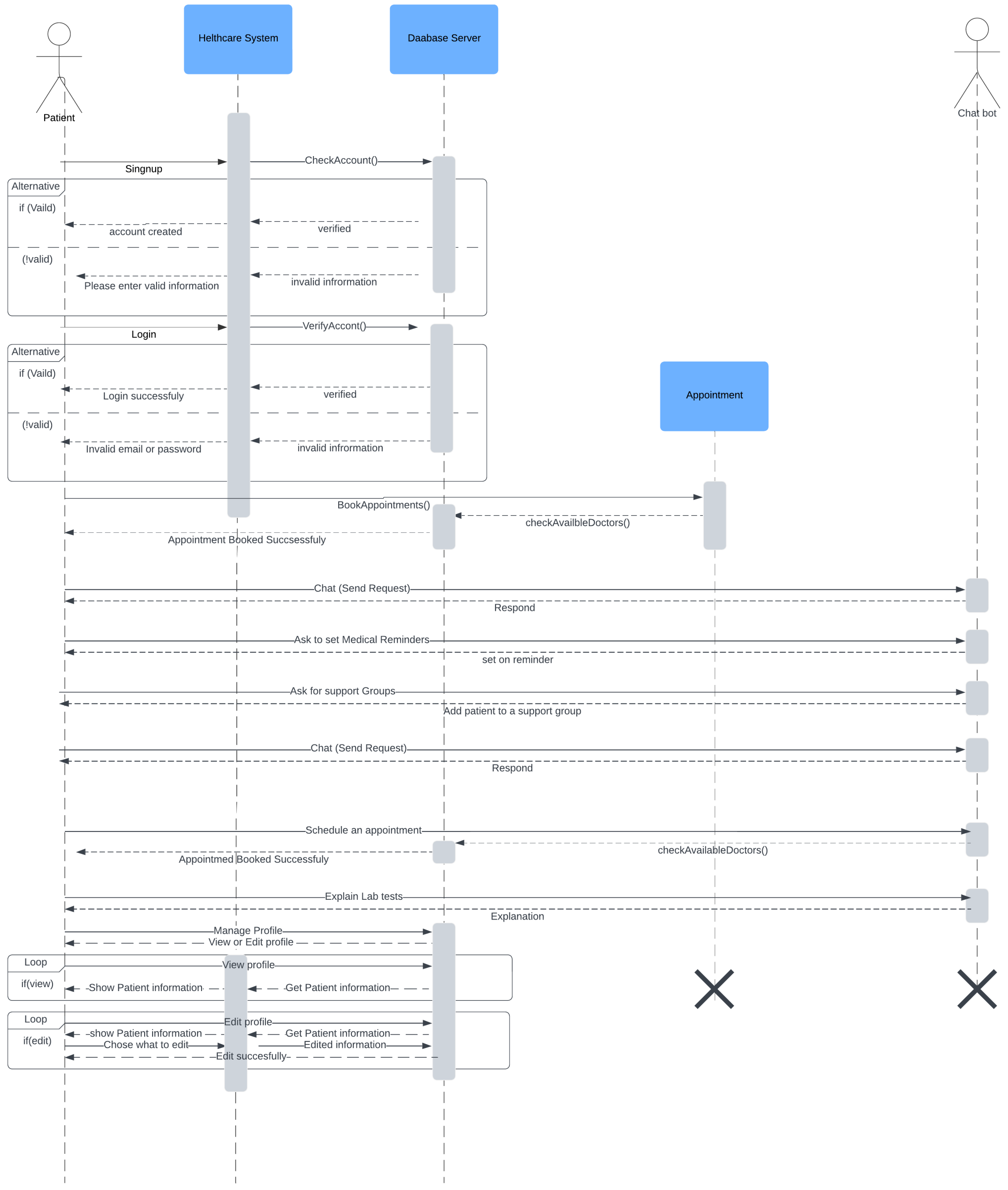


Figure 17: Patient Sequence Diagram

3.7.1 Patient Sequence Diagram

- Signup: patient access the index page and can register on the website then his information is checked and validated in database.

- Login: patient enter his username and password his information is checked and validated in database to make sure that his account is registered to access the patient page
- Book Appointment: patient choose the doctor according to his availability and can book manually or via chatbot
- interaction with chatbot: patient can easily just chat with it or ask fo medical advice, patient can set medical reminders his information is checked and validated in database. via chatbot .
- Patient can have access to manage his profile

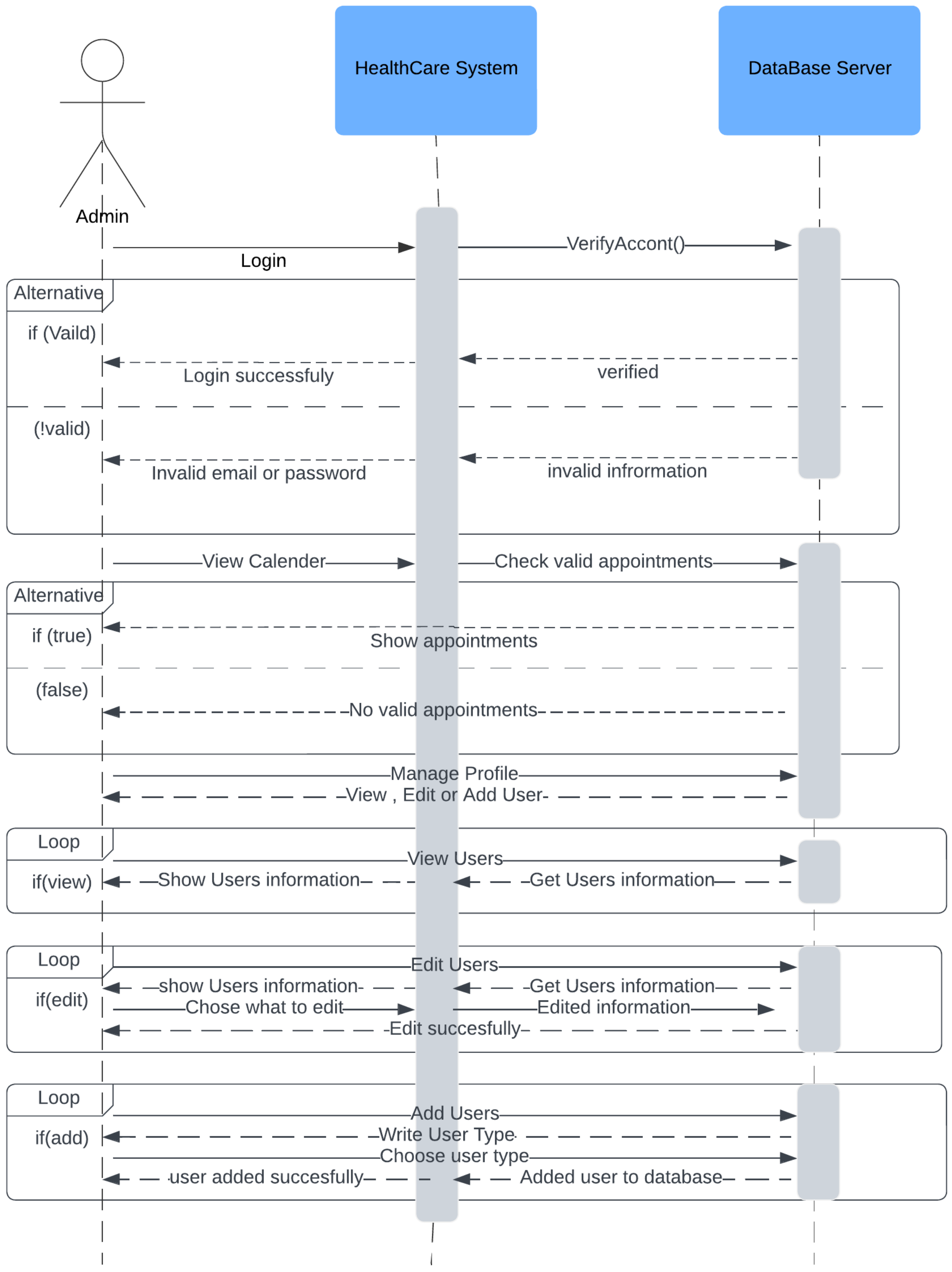


Figure 18: Admin Sequence Diagram

3.7.2 Admin Sequence Diagram

- Login: Admin enter his username and password his information is checked and validated in database to make sure that his account is registered to access the admin page
- View Calender: Admin can easily access his calender to check avaiable appointments and the database checks if the appointments are valid. If the appointments are valid, they are shown to the admin. Otherwise, no valid appointments are displayed.
- Admin can have access to manage users profile. Admin can view all users in database, Edit users admin can edit specific things in each user or Add new users and all this functions are accessed by database.

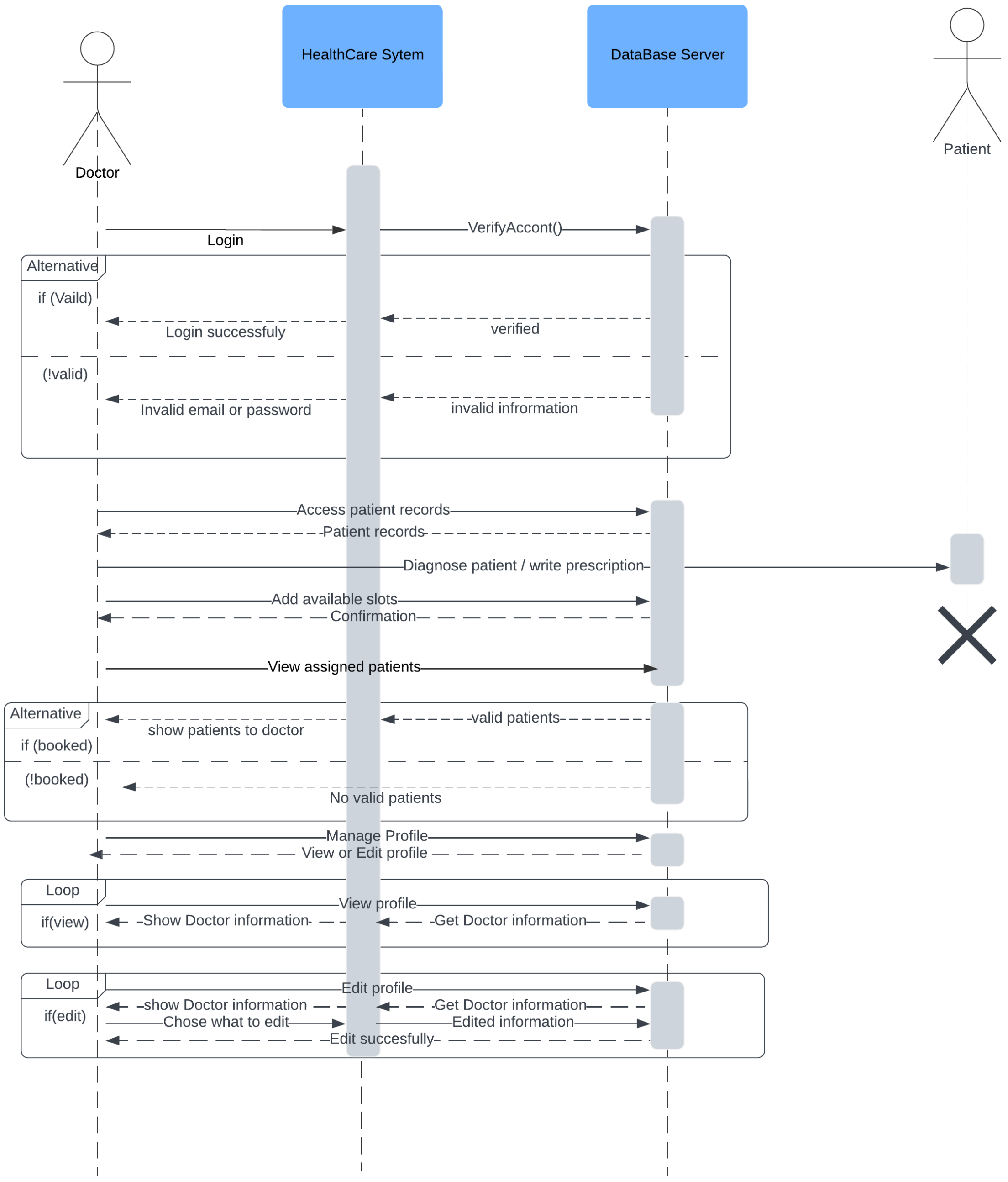


Figure 19: Doctor Sequence Diagram

3.7.3 Doctor Sequence Diagram

- Login: Doctor enter his username and password his information is checked and validated in database to make sure that his account is registered to access the patient page

- Access patient record: Doctor can easily access patient records and write the diagnoses or prescription to the patient.

- View Assigned patients: Doctor can view his assigned patients and the database checks if the patients are valid. If the patients are valid, they are shown to the doctor.Oherwise, no valid patients are displayed.
- Add available slots: Doctor can add his slots and validated in database to confirm that this slot is empty.
- Doctor can have access to manage his profile whether to view his profile or edit his profile.

3.8 Interface viewpoint

4 Data Design

4.1 Data Description

The healthcare system processes and stores data related to patients, doctors, appointments, and medical metrics. This data includes sensitive healthcare information such as personal details, health records, and appointments. It is structured and stored in a relational database for efficient querying and reporting.

- **Patient Data** Includes personal details such as name, contact information, health history. This data is used to manage appointments,access medical information and prescribe medication.
- **Doctor Data** Contains information about doctors, including their specialties, contact details, and appointment schedules.
- **Appointment Data** Tracks the appointments between patients and doctors, including the date, time, and assigned doctor.
- **Profile Management:** Patients can view and manage their personal information, update contact details, and review past appointments.
- **Medical records** Patients can view and upload their medical information,

4.2 Dataset Description

If your project includes the use of a dataset provide a clear description in this section.

4.3 Database design description

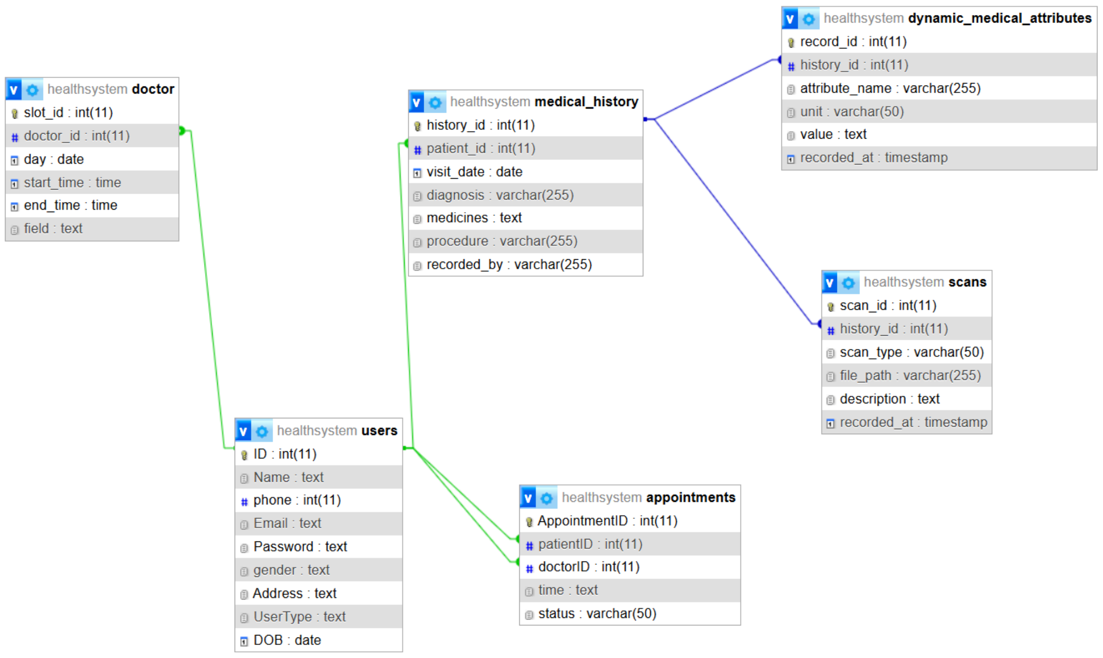


Figure 20: Database Schema

5 Human Interface Design

5.1 User Interface

The user interface of our HealthCare chatbot system is made to be simple and user friendly to provide users the best possible experience. The system is designed to meet the needs of three types of users: Patients , Admin and Doctors , the functionallity is outlines in the following features:

5.1.1 Patient

The patient interface is designed to offer easy experience for patients to organize their health realted tasks.

- **Signup and Login:** Patients can register and log in to the system. Upon succesful login the patient gain access to their profile.
- **Book Appointments:** Patients can view available doctors and schedule appointments based on the doctor's availability. They can choose to book an appointment manually or through the chatbot interface for ease of access.
- **Interaction with Chatbot:** The patient can chat with a built-in chatbot for medical advice, setting reminders for medication, joining support groups , and explaining lab results or medications.

- **Profile Management:** Patients can view and manage their personal information, update contact details, and review past appointments.

5.1.2 Doctor

The Doctor interface is designed to allow them to manage their activites through some functionalities.

- **View assigned patients** Doctors can view the list of patients assigned to them.
- **Manage appointments** Doctors can view their scheduled appointments or add new slots for appointments.

5.1.3 Admin

The Admin interface provides comprehensive control over the sustem, ensuring proper management and monitoring of patients , doctors and appointments through some functionalities.

- **User managment** Admin can manage both patient and doctor accounts.
- **Monitor appointments** Admin can view all appointments.

5.2 Screen Images

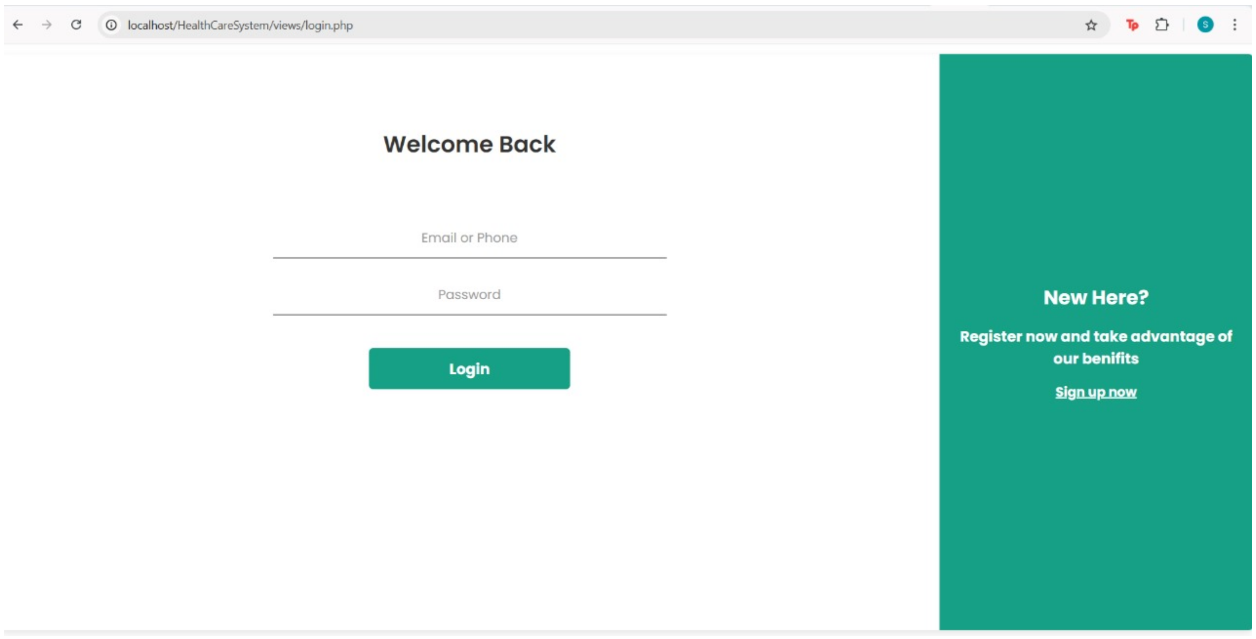

localhost/HealthCareSystem/views/login.php
Welcome Back
Email or Phone
Password
Login
New Here?
Register now and take advantage of our benifits
Sign up now

Figure 21: login

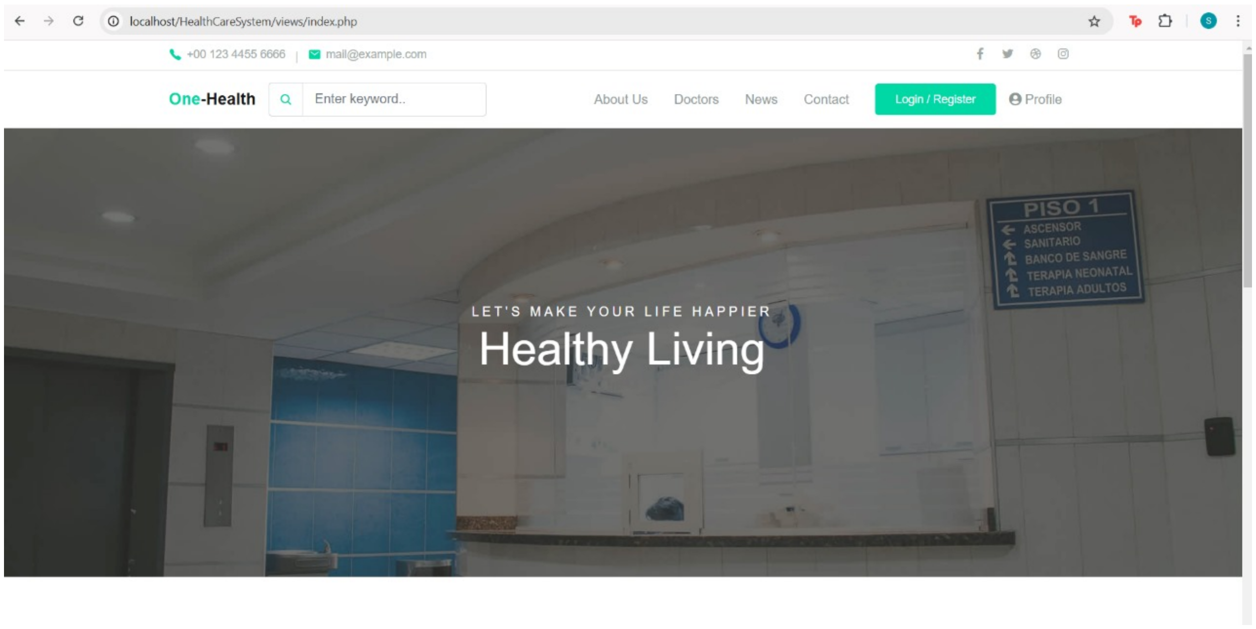
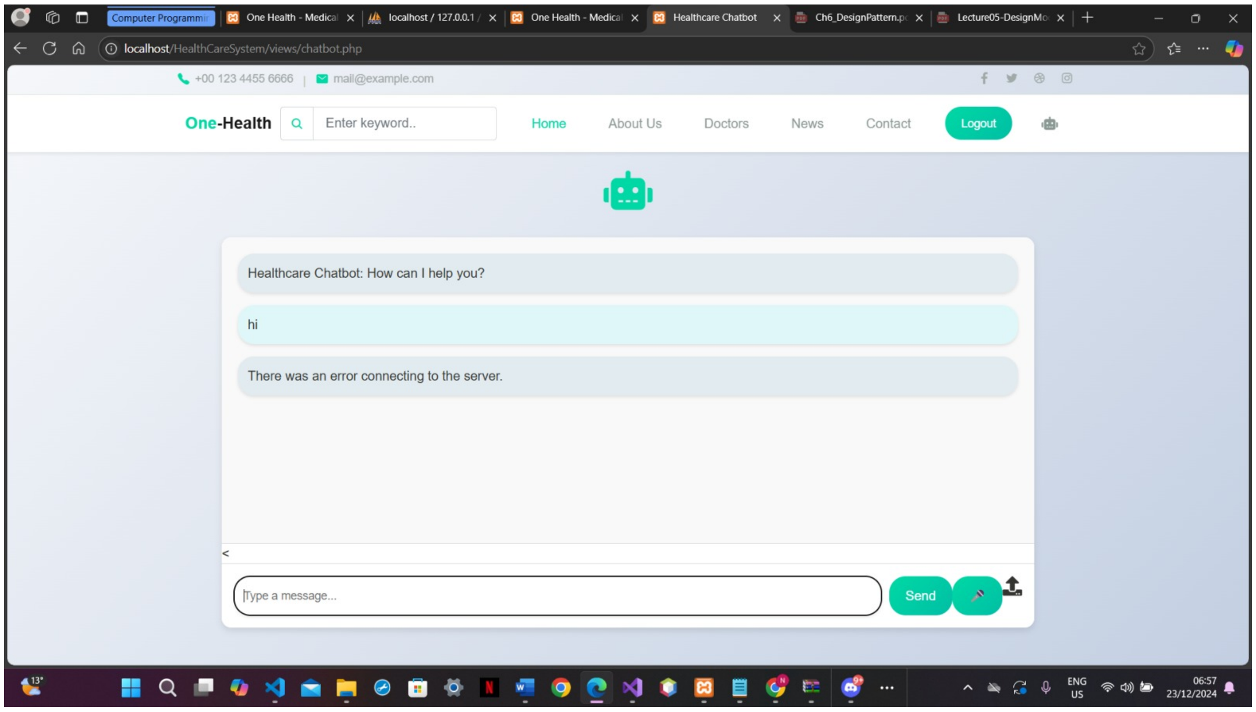


Figure 22: frontend website

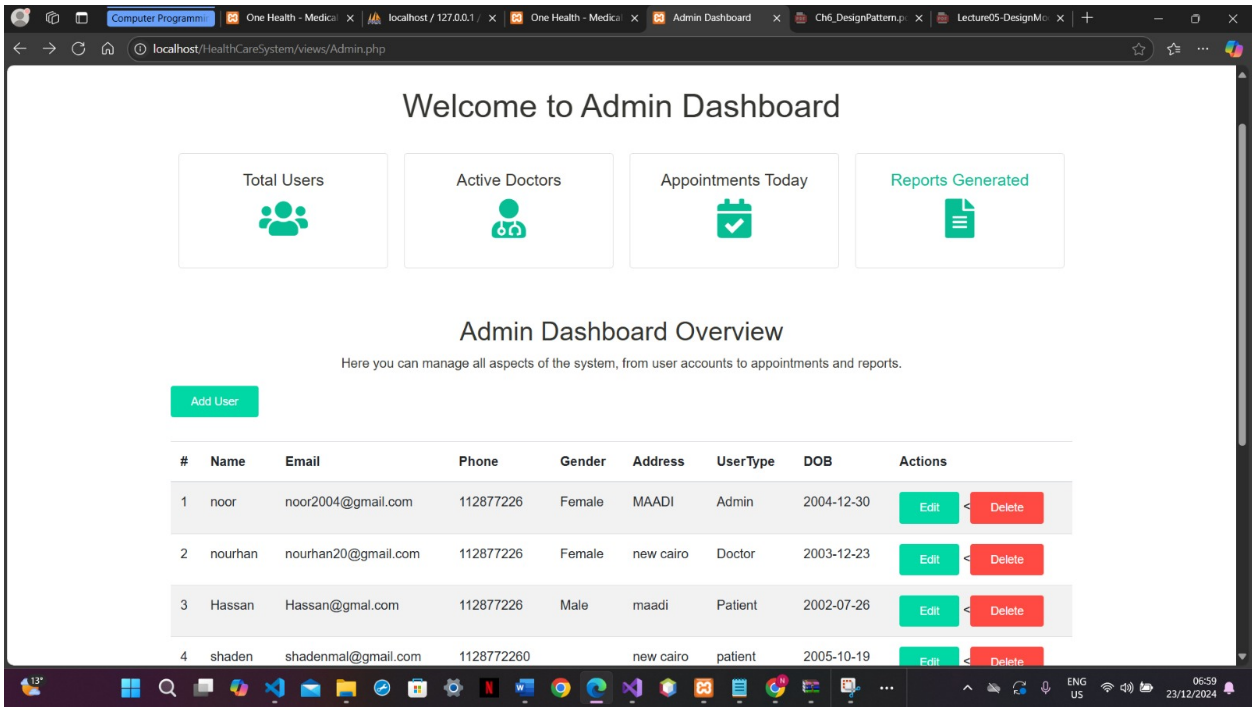


Healthcare Chatbot: How can I help you?

hi

There was an error connecting to the server.

Figure 23: Chatbot



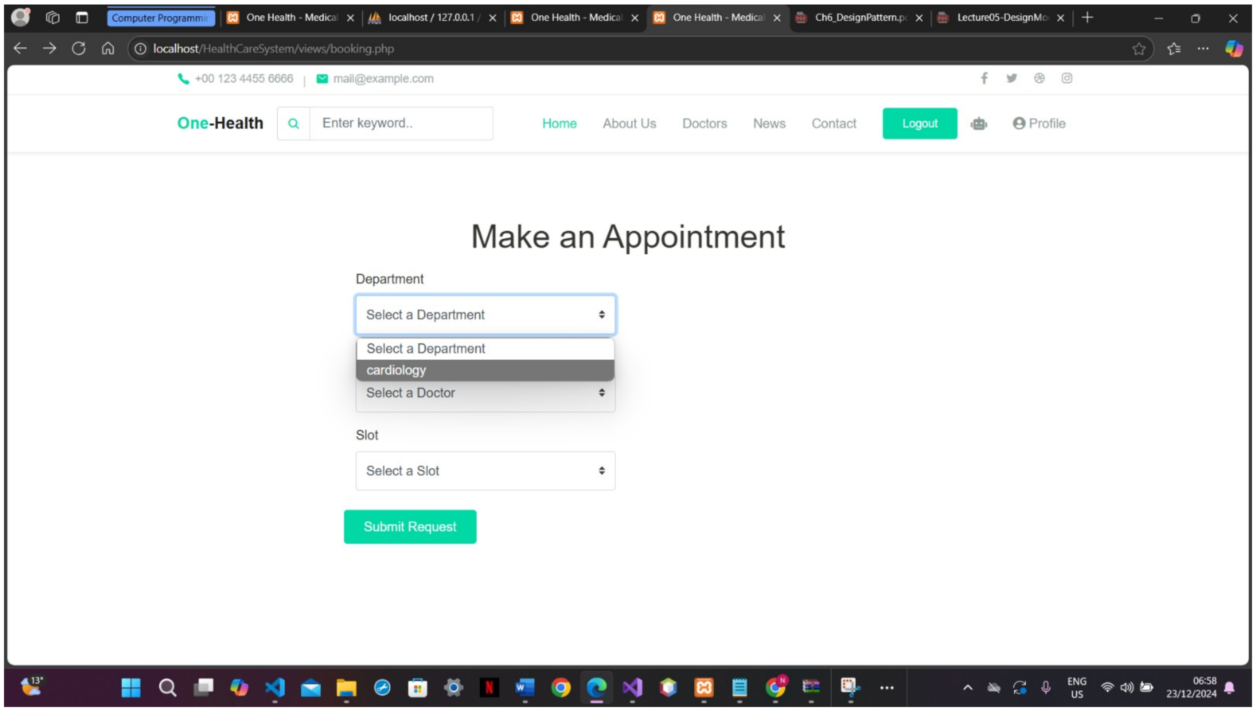
Welcome to Admin Dashboard

Admin Dashboard Overview

Here you can manage all aspects of the system, from user accounts to appointments and reports.

#	Name	Email	Phone	Gender	Address	UserType	DOB	Actions
1	noor	noor2004@gmail.com	112877226	Female	MAADI	Admin	2004-12-30	Edit Delete
2	nourhan	nourhan20@gmail.com	112877226	Female	new cairo	Doctor	2003-12-23	Edit Delete
3	Hassan	Hassan@gmal.com	112877226	Male	maadi	Patient	2002-07-26	Edit Delete
4	shaden	shadenmal@gmail.com	1128772260		new cairo	patient	2005-10-19	Edit Delete

Figure 24: Admin dashboard



Make an Appointment

Department: Select a Department

Select a Department
cardiology

Select a Doctor

Slot: Select a Slot

Figure 25: Make an appointment

5.3 Screen Objects and Actions

This section lists the screen objects and the actions associated with them in our healthcare system:

- **Form pages:**
  - e.g. Login , Sign up , include form fields covering needing data for login and sign up and a submit button.
- **Appointments Form:**
  - include doctor and slot drop down to allow patient to choose the doctor and available slot.
- **Chatbot interface:**
  - include message input box to allow patient to type their input , image upload to upload images of lab results to explain it, text to speech button and to join support groups.
- **Profile Managment:**
  - include edit , delete buttons to allow patient and doctor to update their info.
- **Doctor Dashboard:**
  - **Add Slot form:**Allows the doctor to create available appointment slots by entering the start date , end date and day for new bookings.
  - **View patient schedule:** Displays a schedule of patients assigned by doctor.
- **Admin Dashboard:**
  - **Reports:**
  - **Calender:** Displays a calender showing all appointments for both doctor and patients.
  - **Active Doctors:** Displays all available doctor in our system in a tabular form.
  - **Total users:** Displays all users in our system in tabular form.

6 Requirements Matrix

Provide a cross reference that traces components and data structures to the requirements in your SRS document. float

Table 17: Requirements Ratrix

Req. ID	Req Desc	Class	Test Cases ID	Status
A01	Admin adds, edits, deletes and views patients	User	TC01	In Progress
A02	Admin adds, edits, deletes and views doctors	User	TC02	In progress
P01	Patient registers and logs in	User	-	Done
P02	Patient can book appointments through form and through the chatbot	User	-	Done
P03	Patient can view scheduled appointments in profile	User	-	Done
P04	Patient can edit details in profile	User	-	Done
D01	Doctor can log into account and log out	User	-	Done
D02	Doctor can add available time slots for booking	User	-	Done
D03	Doctor can edit personal information in profile	User	-	Done

7 APPENDICES

Appendices may be included, either directly or by reference, to provide supporting details that could aid in the understanding of the Software Design Document.

7.1 Github

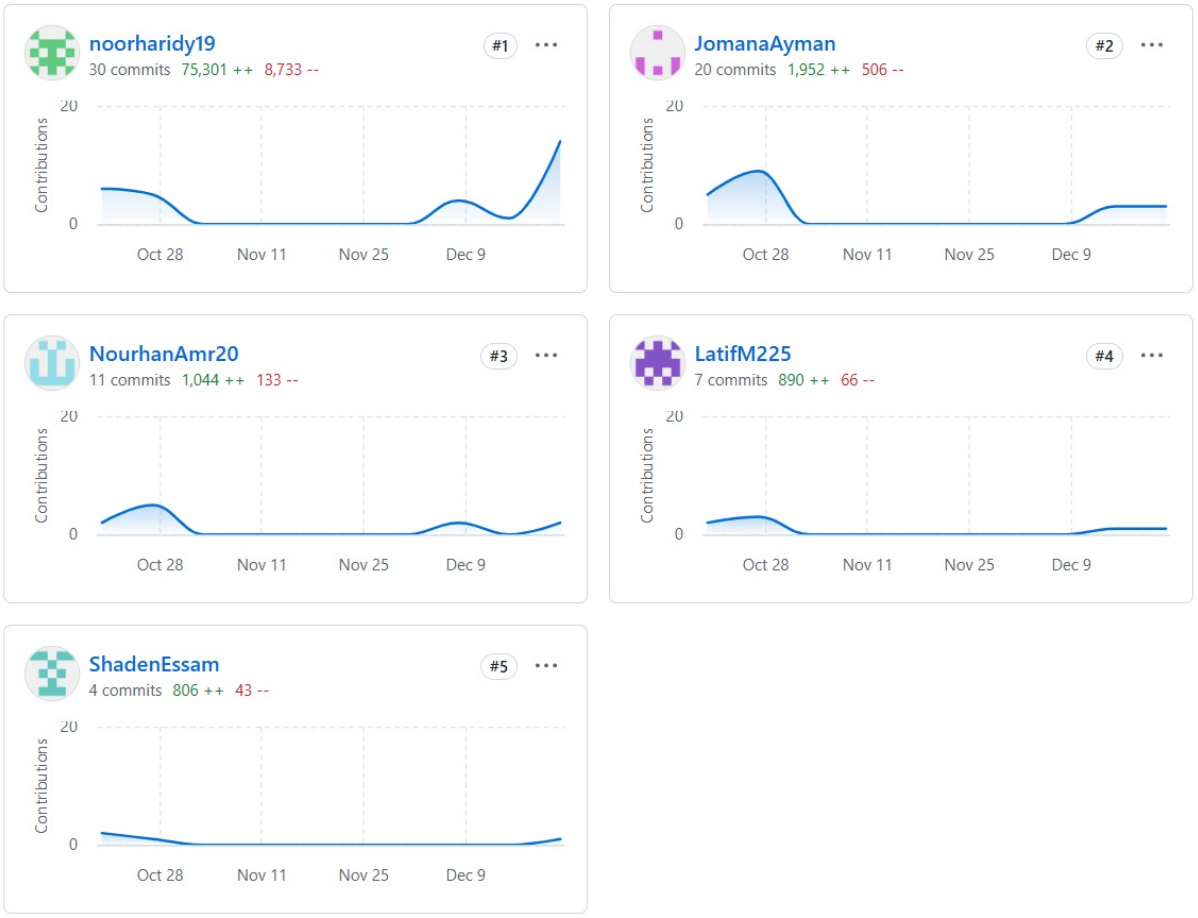
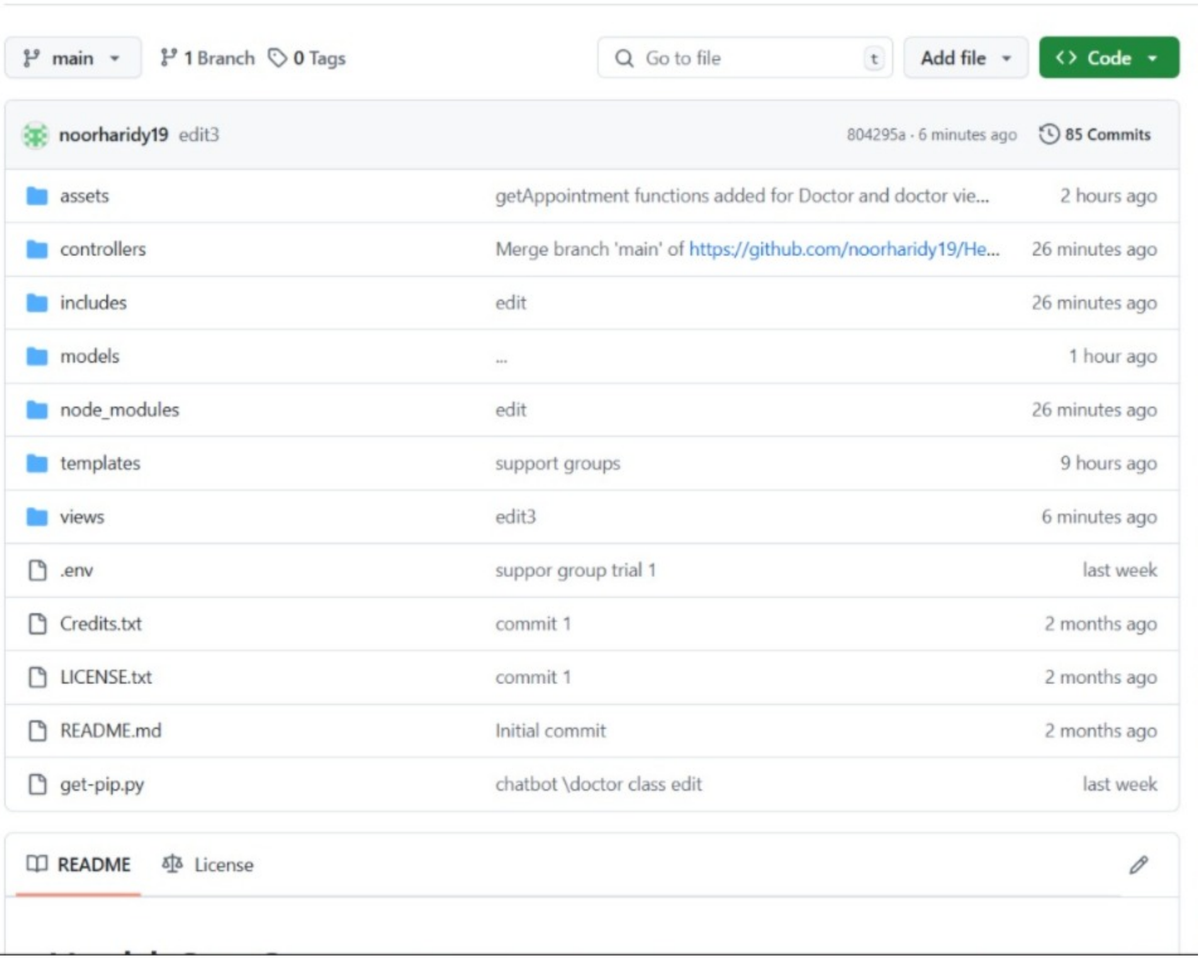


Figure 26: Commits in our project



noorharidy19 edit3 804295a · 6 minutes ago 85 Commits

assets	getAppointment functions added for Doctor and doctor vie...	2 hours ago
controllers	Merge branch 'main' of https://github.com/noorharidy19/He...	26 minutes ago
includes	edit	26 minutes ago
models	...	1 hour ago
node_modules	edit	26 minutes ago
templates	support groups	9 hours ago
views	edit3	6 minutes ago
.env	suppor group trial 1	last week
Credits.txt	commit 1	2 months ago
LICENSE.txt	commit 1	2 months ago
README.md	Initial commit	2 months ago
get-pip.py	chatbot \doctor class edit	last week

Figure 27: our reprisotery

7.2 Other appendices as appropriate

Optional section.