

**Epoka University**

**Faculty of Engineering and Architecture**

**Department of Computer Engineering**

**CEN302 – Software Engineering**

**HMS Requirements Specification**

**May 29, 2025**

**Prepared by:**

**Alban Plakalliu**

**Erlis Ciko**

**Gleris Xhoxha**

**Klei Qorri**

**Deni Mekollari**

**Erblin Boletini**

***Received by: Msc. Shkumbin Fida, Msc. Edlira Cani***

Table of Contents

[**1.** **Executive Summary** 4](#_Toc199455646)

[**1.1** **Project Overview** 4](#_Toc199455647)

[**2.** **Product/Service Description** 5](#_Toc199455648)

[**2.1 Product Context** 5](#_Toc199455649)

[**2.2 User Characteristics** 5](#_Toc199455650)

[**2.1** **Assumptions** 6](#_Toc199455651)

[**2.2** **Constraints** 6](#_Toc199455652)

[**2.3** **Dependencies** 7](#_Toc199455653)

[**3.** **Requirements** 8](#_Toc199455654)

[**3.1** **Functional Requirements** 8](#_Toc199455655)

[**3.2 Non-Functional Requirements** 10](#_Toc199455656)

[**3.2.1 Product Requirements** 10](#_Toc199455657)

[**3.2.2 Organizational Requirements** 12](#_Toc199455658)

[**3.2.3 External Requirements** 13](#_Toc199455659)

[**3.3 Domain Requirements** 14](#_Toc199455660)

[**4. Software Design/Diagrams** 15](#_Toc199455661)

[**4.1 Requirement Analysis** 15](#_Toc199455662)

[**4.1.1 User Scenarios** 15](#_Toc199455663)

[**4.1.2 User Cases** 20](#_Toc199455664)

[**4.2** **Behavioral Diagrams** 27](#_Toc199455665)

[**4.2.1 Use Case Diagrams** 27](#_Toc199455666)

[**4.2.2 Activity Diagrams** 31](#_Toc199455667)

[**4.2.3 ER Diagram** 46](#_Toc199455668)

[**4.2.4 Class Diagrams** 47](#_Toc199455669)

[**4.2.5 Sequence Diagrams** 49](#_Toc199455670)

[**4.2.6 Components Diagram** 53](#_Toc199455671)

[**4.2.7 State Diagram** 54](#_Toc199455672)

[**4.2.8 Project Planning** 55](#_Toc199455673)

[**5. Technologies Used** 57](#_Toc199455674)

[**5.1 Implementation Images** 57](#_Toc199455675)

# **Executive Summary**

## **Project Overview**

The Hospital Management System (HMS) aims to streamline administrative and medical operations within a medical facility. The system will facilitate patient record management, appointment scheduling, and secure communication between medical professionals and patients. This system aims to reduce paperwork, streamline administrative processes, and ensure secure data handling for medical institutions. Furthermore, the system will comply with relevant healthcare regulations to maintain data integrity and confidentiality.

This specification defines the requirements and functionalities of the Hospital Management System, serving as a foundational document for developers, stakeholders, and system testers. The purpose of this specification is to outline the necessary features and constraints of the system, ensuring alignment with user needs and regulatory compliance

**In Scope**:

* **Role-Based User Management**  
  Implement secure authentication and authorization so that Patients, Doctors, and Managers each have tailored access.
* **Appointment Scheduling & Notifications**  
  Provide a full appointment lifecycle: patients can request or book slots with recommended (or chosen) doctors; doctors can accept, reschedule, or cancel; and both parties receive real-time notifications and calendar integration to minimize no-shows.
* **A comprehensive dashboard** for different user types, ensuring that every stakeholder has access to the tools they need.
* **AI Integration** for doctor recommendation based on the symtopms that the user shows.

**Out of Scope**

The following features are not covered in this phase of the project and may be considered for future iterations:

* Integration with third-party insurance providers for automated billing and claims processing.

# **Product/Service Description**

The Hospital Management System simplifies patient management, scheduling, and communication within medical facilities. It provides a web-based and mobile-accessible platform integrating with hospital management systems for real-time updates. Security measures ensure compliance with industry standards, protecting sensitive medical data. Future enhancements may include AI-based diagnostics and expanded telemedicine support.

## **2.1 Product Context**

The Hospital Management System is a standalone platform designed to function independently while also integrating with existing healthcare infrastructure. It connects with hospital databases, appointment scheduling systems, and secure communication channels to streamline patient care. The system ensures seamless data exchange, allowing interoperability with electronic health record (EHR) systems, laboratory databases, and insurance verification portals. Through well-defined APIs, it can integrate with third-party applications, enabling extended functionalities such as automated reporting and analytics.

## **2.2 User Characteristics**

The Hospital Management System is designed for various user groups, each with distinct roles and levels of expertise.

**Doctor**

Can access and manage patient medical records.

Can update appointment details.

Can view visit summaries.

Can use system dashboards for workflow management.

Can log timestamps for patient interactions and procedures.

**Patient**

Can view personal health records and visit history.

Can schedule an appointment in a preferred medical field.

Can use the hospitals services.

Gets personalized advice from AI for choosing the right hospital department.

**Manager**

Can monitor active users and doctors

Can add users and doctors

Can change user passwords

Can delete users

Can assign users to doctors

## **Assumptions**

* Several assumptions have been made to guide the development of the Hospital Management System:
* Users will have access to a stable internet connection to utilize the system’s web-based and mobile functionalities.
* The system will be deployed on secure cloud-based infrastructure to ensure scalability and data protection.
* Medical staff and administrative personnel will receive basic training to use the system effectively.
* The system will operate on commonly used web browsers and mobile devices without requiring additional software installations.
* Hospitals and medical centers using the system will comply with industry regulations to ensure data security and patient privacy.

## **Constraints**

The design of the Hospital Management System will be influenced by several constraints that impact its functionality, security, and performance. These constraints are as follows:

Parallel Operation with an Old System

The new system must be able to run parallel with the existing legacy system for a transition period. This will allow staff to adjust to the new system while ensuring the old system remains operational for any unforeseen issues during the switchover. The new system must support data synchronization and integration with the old system, minimizing disruption in daily operations.

Audit Functions

Given the sensitive nature of medical data, the system must implement comprehensive audit functions. This includes maintaining an audit trail that logs all user actions, such as patient data access, modifications, and report generation. Logs should be stored securely and be easily accessible for auditing purposes. These logs will help ensure compliance with regulations such as HIPAA and GDPR.

Access, Management, and Security

The system will handle confidential patient information and must adhere to stringent access control policies. Only authorized personnel will be granted access to different levels of the system based on roles (e.g., doctors, nurses, administrators). Authentication mechanisms, such as multi-factor authentication, will be enforced. Additionally, data encryption both at rest and in transit will be implemented to protect sensitive medical data.

Criticality of the Application

As the system is integral to daily operations in a medical center, its failure could result in significant operational disruptions. The system must be highly available, with backup mechanisms, failover strategies, and disaster recovery plans in place. A high level of reliability and uptime is critical to ensure continuous patient care and avoid interruptions to medical services.

System Resource Constraints

The system will operate within the constraints of the available hardware and infrastructure. Disk space and memory are limited, so the design must be optimized to minimize resource consumption. Database management should be efficient, especially when handling large volumes of patient data. The system should also be scalable to accommodate growth in both data and user traffic over time.

Other Design Constraints

Programming Language and Framework: The system will be developed using modern and widely accepted programming languages and frameworks, ensuring compatibility with existing IT infrastructure. The backend will be built using ExpressJs, while the frontend will utilize React for a responsive user interface.

Design Standards: The system will follow established software engineering principles, such as modularity, maintainability, and adherence to coding standards. This will ensure long-term system sustainability and ease of updates or modifications.

## **Dependencies**

Receptionist-Centric Dependencies

* Adding New Patient/Doctor: Cannot be done if the receptionist is unavailable or not registered.
* Deleting Patient/Doctor: Not allowed if the receptionist is unavailable.
* Doctor-Patient-Centric Dependencies
* Viewing Profiles/Records: Doctors can only view records of patients assigned to them.
* Viewing Latest Medical Visit: Patients can only view their latest record after the doctor saves it.
* Examination: Cannot take place unless a doctor is assigned to the patient.
* Cross-Actor Dependencies

# **Requirements**

## **Functional Requirements**

In the example below, the requirement numbering has a scheme - BR\_LR\_0## (BR for Business Requirement, LR for Labor Relations). For small projects simply BR-## would suffice. Keep in mind that if no prefix is used, the traceability matrix may be difficult to create (e.g., no differentiation between '02' as a business requirement vs. a test case)

The following table is an example format for requirements. Choose whatever format works best for your project.

For Example:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Req# | Requirement | Comments | Priority | Date Rvwd | SME Reviewed / Approved |
| FR\_01 | The system should allow patients to book appointments. | Essential feature for patient convenience and system efficiency. | 2 | 17/03/25 | Alban Plakalliu / Erlis Ciko |
| FR\_02 | Doctors should be able to view and update patient medical records securely. | Critical for doctors to provide effective care; security is paramount. | 2 | 17/03/25 | Klei Qorri / Gleris Xhoxha |
| FR\_03 | The receptionist should be able to add a new patient. | Important for the recording of patients. | 1 | 18/03/25 | Deni Mekollari / Erblin Boletini |
| FR\_04 | The system should include a dashboard to show the records for each patient. | So, you have opportunities to check the appointments or patient data. | 1 | 18/03/25 | Gleris Xhoxha / Erlis Ciko |
| FR\_05 | Users should log in using secure authentication. | Ensures secure access to the system, protecting sensitive data. | 2 | 18/03/25 | Erblin Boletini / Alban Plakalliu |
| FR\_06 | Users should be able to reset their password securely. | Helps maintain account access and user autonomy. | 2 | 19/03/25 | Erlis Ciko / Klei Qorri |
| FR\_07 | Users should be able to view and update their personal profile information. | Allows patients and staff to keep their contact and medical details current. | 2 | 20/03/25 | Gleris Xhoxha / Deni Mekollari |
| FR\_08 | Users should be able to view their scheduled appointments. | Enhances transparency and planning for both patients and doctors. | 1 | 24/03/25 | Klei Qorri / Alban Plakalliu |
| FR\_09 | Each doctor should be able to view only their list of patients. | Maintains data privacy and relevance per doctor role. | 2 | 25/03/25 | Erlis Ciko / Gleris Xhoxha |
| FR\_10 | Doctors should be able to add new medical records to a patient profile. | Crucial for ongoing treatment and documentation. | 1 | 26/03/25 | Alban Plakalliu / Deni Mekollari |
| FR\_11 | Doctors should be able to download a patient's medical record. | Enables offline review, referrals, or sharing when needed. | 2 | 27/03/25 | Gleris Xhoxha / Klei Qorri |
| FR\_12 | Managers should be able to add a new staff member to the system. | Necessary for keeping the system updated with current human resources. | 2 | 28/03/25 | Erblin Boletini / Deni Mekollari |
| FR\_13 | Managers should be able to delete a user account. | Needed for data cleanup and removing inactive or unauthorized accounts. | 2 | 31/03/25 | Alban Plakalliu / Gleris Xhoxha |
| FR\_14 | The system should allow staff to mark an appointment/reservation as concluded. | Ensures records are updated and statuses reflect real-world operations. | 1 | 01/04/25 | Klei Qorri / Erblin Boletini |
| FR\_15 | Users should be able to log out securely from the system. | Important for maintaining session security and integrity | 1 | 03/04/25 | Erblin Boletini / Erlis Ciko |
| FR\_16 | Users should be able to filter doctors by department. | Important for ease of access of doctor by user | 1 | 04/04/25 | Erblin Boletini |
| FR\_17 | Users can ask an AI model to automatically recommend department | Ease of access for user experience | 2 | 20/05/25 | Erlis Ciko |
| FR\_18 | User can see and select free dates for a specific doctor. | Improved user experience | 1 | 12/04/25 | Deni Mekollari/Gleris Xhoxha |
| FR\_19 | User can select timeslot after selecting appointment date. | Better granularity for bookings | 1 | 15/04/25 | Deni Mekollari/Alban Plakalliu |
| FR\_20 | User can select multiple services per appointment. | Improved service availability for the user | 2 | 22/04/25 | Alban Plakalliu/Klei Qorri |

## **3.2 Non-Functional Requirements**

### **3.2.1 Product Requirements**

#### **3.2.1.1 User Interface Requirements**

The Medical Center Management System is expected to meet several non-functional criteria that ensure reliability, usability, and performance. First and foremost, the user interface should be responsive, meaning it must adapt seamlessly to various screen sizes, whether accessed from desktop computers, tablets, or mobile devices. This ensures that all users, regardless of the device used, can navigate and interact with the system comfortably.

Moreover, the user experience and functionality should remain consistent across different web browsers. Whether the system is accessed through Google Chrome, Mozilla Firefox, Safari, Microsoft Edge, or any other modern browser, users should encounter the same layout, behavior, and feature set.

The system must also be intuitive and easy to grasp, especially for new users who may not be technologically advanced. Navigating the system should require minimal learning, and features should be self-explanatory and well-organized to support fast adoption.

Additionally, the system should support secure access at any time and from any location. Registered users must be able to log in using their confidential credentials (username and password) regardless of their geographical location or time zone. This functionality is essential for doctors, patients, and staff who need access outside standard working hours.

From a performance perspective, the system should be efficient and fast in processing user operations such as login, appointment booking, medical record retrieval, and data updates. The overall latency of the system should primarily depend on three factors: the user’s internet connection, the efficiency of the algorithms used in the system's backend, and the size of the database. However, since the system is not expected to have an enormous volume of users in its initial stages, the database size is unlikely to impact performance significantly.

Finally, to ensure cost-efficiency, reliability, and availability, the system will be deployed on Amazon Web Services (AWS). This cloud platform provides robust infrastructure, scalable resources, and high uptime, all of which are crucial for ensuring uninterrupted access and reliable operation of the Hospital Management System.

#### **3.2.1.2 Learnability**

Receptionists and doctors should be able to master the system within a few hours due to its simple and intuitive interface. Patients will not require any specific training as the system is designed to be user-friendly. This documentation can also serve as a quick user guide for receptionists and doctors of the polyclinic. In case of any error, a clear and detailed message will be shown to help the user understand what went wrong. Each user remains responsible for their own actions within the system.

#### **3.2.1.3 Accessibility**

Each user is assigned a unique username and password when they are registered in the system. Both receptionists and doctors who work at the polyclinic have access to the system. Patients registered with the polyclinic can also access the system using their credentials. All registered users are able to log in and use the system at any time and from any location, ensuring continuous and flexible access to necessary medical and administrative services.

#### **3.2.1.4 Efficiency**

Each operation within the system is designed to be fast and occur in real time, providing a smooth user experience. Once users become familiar with the system, they will be able to complete tasks and perform each operation within minutes, ensuring efficient workflow for all users.

#### **3.2.1.5 Memorability**

The system is designed to be intuitive, so even if users only vaguely remember how to use it, they will quickly regain familiarity. If users return to the system after a period of inactivity, they can expect to re-establish their proficiency within the first hour. The user interfaces are crafted to be easily understandable, utilizing clear pictures, icons, buttons, and descriptive elements to guide the user effectively.

#### **3.2.1.7 Errors**

The system aims to reduce the error rate compared to the current processes in place. For every entry involving sensitive data, a double-check procedure will be implemented, requiring users to confirm the accuracy of the information before submission. In case an error occurs, users will have the ability to edit and correct the data immediately, minimizing the impact of mistakes and maintaining data integrity.

#### **3.2.1.8 Capacity**

This application is designed to meet all the needs of a single polyclinic, supporting simultaneous use by patients, doctors, and receptionists. It operates in real time, so any changes made by a user will be immediately reflected to others based on their access permissions. Since all users share the same database, multiple simultaneous requests may cause slight delays as queries are processed. To mitigate this, user changes are first stored locally on their devices before being sent to the database, allowing users to continue working smoothly even if delays occur. The application will be hosted on a web server, and both the application and its database are expected to require only minimal storage space.

### **3.2.2 Organizational Requirements**

#### **3.2.2.1 Availability**

The system will be available 24 hours a day, 7 days a week, ensuring continuous access for all users including patients, doctors, and receptionists. Since the system is web-based, any registered user with internet access can log in and manage their tasks from any location. Given the sensitive and time-critical nature of medical data, the application will be designed to minimize downtime and maintain high availability. Scheduled maintenance will be carefully planned to avoid disruption, and in the case of any unexpected issues, unscheduled maintenance will ensure that downtime does not exceed one hour.

#### **3.2.2.2 Latency**

The response time of the system depends primarily on the user’s internet connection, the size of the medical records database, and the efficiency of the backend algorithms used to retrieve and update data. The system architecture and code are optimized to reduce delays and provide real-time updates, ensuring a smooth user experience even during peak usage.

#### **3.2.2.3 Monitoring**

The system will be developed with a focus on security and reliability. User interfaces are designed for ease of use and robustness to prevent crashes. Maintenance personnel will receive periodic system reports to monitor performance, detect potential issues early, and identify opportunities for improvement. Input validation will be implemented throughout the system—for example, during login, patient registration, staff addition, profile updates, and password changes. Users will receive clear and informative messages whenever their input fails validation or when errors occur, facilitating quick resolution

#### **3.2.2.4 Maintenance**

The application will utilize MySQL as the database backend and run on an Apache web server, both of which provide stable and well-supported platforms for maintenance. The system is modularly designed, enabling easy addition of new features and updates without affecting existing functionality. In case of system malfunction, a simple restart of the application or server will be attempted first. If issues persist, the maintenance team will be notified for further investigation. Users will be informed ahead of time about any software updates or bug fixes that could impact system availability.

#### **3.2.2.5 Operations**

Users will be able to log in securely at any time to access their personal information and medical data. Access controls ensure that sensitive information is only visible to authorized personnel, such as doctors viewing their own patients’ records or managers updating examination fees. The system will support CRUD (Create, Read, Update, Delete) operations for doctors, patients, medical services, and appointments. Doctors will be able to create new medical examination records, view and download patient profiles and medical reports in PDF format, and share these records securely with other healthcare providers when necessary. Patients will be able to book, cancel, and view their appointments, contact doctors via email, and leave feedback on the service provided. The system will also offer monthly statistical reports accessible only to the polyclinic director. While the system is accessible 24/7, the busiest usage hours are expected to be between 8 AM and 11 AM, and 4 PM and 7 PM, reflecting typical patient visit patterns. All user passwords and sensitive data will be securely protected using hashing and encryption mechanisms.

#### **3.2.2.6 Standards Compliance**

The Hospital Management System is designed to fully comply with existing healthcare and data protection standards. While the system is a new digital solution for managing patient and staff records within a polyclinic, it ensures that all personal and medical information fields conform to the official formats currently used by the polyclinic. User IDs and other critical identifiers will follow the standards mandated by the Ministry of Health of Albania. The system’s development and operation strictly adhere to relevant laws and regulations to guarantee legal compliance and protect patient data integrity.

#### **3.2.2.7 Portability**

Being a web-based application, the system will function uniformly across all operating systems, including Windows, macOS, Linux, Android, and iOS. Users will require only an internet-enabled device such as a computer, tablet, or mobile phone, along with an internet connection, to access the system anywhere and anytime without compatibility issues.

### **3.2.3 External Requirements**

#### **3.2.3.1 Security**

The system handles highly sensitive personal and medical information, necessitating strong security measures. Compliance with Albanian law No.9887 (dated 10.03.2008) and its amendment No.48/2012, “On the Protection of Personal Data,” is strictly enforced. Personal data within the system is kept confidential and accessible only to authorized users such as doctors, receptionists, and patients themselves. Passwords and other sensitive credentials are securely hashed to prevent unauthorized access. Users with access rights are expected to responsibly manage patient information in accordance with privacy laws.

#### **3.2.3.2 Protection**

To safeguard against unauthorized or accidental data access, alteration, disclosure, or destruction, the system implements several protection mechanisms:

* Sensitive information like passwords is encrypted using secure hashing methods.
* User activities are logged and monitored to ensure accountability and traceability.
* Receptionists are responsible for verifying the accuracy of the personal data they enter; the system itself does not verify authenticity.
* All data inputs undergo validation to prevent the insertion of harmful characters or invalid information.
* Critical user actions are confirmed through popup dialogues to prevent unintended changes.
* Patients can only view their own records, ensuring strict data isolation.
* Doctors are restricted to viewing only their own patients’ information to maintain confidentiality.

#### **3.2.3.2 Authorization and Authentication**

The system enforces a robust authentication and authorization framework:

* User authentication requires a username and password.
* Authorization controls restrict access to data and functions based on the user’s role (patient, doctor, or manager).
* Sessions maintain secure and continuous access for logged-in users.
* Failed login attempts trigger clear error messages to inform the user of incorrect credentials.

## **3.3 Domain Requirements**

This web application operates within the Albanian public medical system domain, targeting the digital transformation of patient record management in polyclinics. Its primary goal is to facilitate efficient and quick access to patient data while ensuring stringent security measures to protect sensitive information. The system is intended for use within a single polyclinic environment, operating on a private network without integration to external systems. This containment helps maintain data privacy and simplifies compliance with local regulations and operational policies.

# **4. Software Design/Diagrams**

## **4.1 Requirement Analysis**

### **4.1.1 User Scenarios**

#### **4.1.1.1 User Scenarios List**

|  |  |  |
| --- | --- | --- |
| No. | Name | Description |
| US\_1 | User logs in the system. | User (Patient,Doctor,Manager) logs into the system using their username and password. |
| US\_2 | View personal record. | Patient views personal general information. |
| US\_3 | View list of doctors. | Patient is shown the list of doctor for all departments. |
| US\_4 | Edit personal information. | User can edit his personal information. |
| US\_5 | Ask AI for recommendation. | User can ask AI for department recommendation based on issue. |
| US\_6 | Create a medical appointment. | User creates a new medical appointment with a doctor. |
| US\_7 | Download patient medical record. | Doctor downloads the details of the patient in an appointment. |
| US\_8 | Cancel a reservation. | User cancels an upcoming or existing reservation. |
| US\_9 | Mark appointment as complete. | Doctor marks an appointment as completed. |
| US\_10 | View history of medical records. | Patient views a list of his past records. |
| US\_11 | View Assigned Patients List | Doctor views a list of their assigned patients. |
| US\_12 | View Staff List | Manager views the list of all staff members. |
| US\_13 | Edit assigned patient medical record | Doctor edits his assigned patients medical record. |
| US\_14 | Add a staff member. | Manager adds a new staff member to the system. |
| US\_15 | Delete a user or a doctor. | Manager deletes a patient or staff account. |
| US\_16 | Change profile picture | Users can change their profile picture for a more descriptive profile |
| US\_17 | Filter by department | User can filter the doctors by department |
| US\_18 | Change password | Users can change their password. |
| US\_19 | Assign personal doctor. | Manager can assign personal doctors |
| US\_20 | Log out of the system | User logs out of the system. |

#### **4.1.1.2 User Scenarios Extended**

**US\_01: User logs in the system**

Steps:

1. User enters their username and password based on their role.
2. User presses the login button.
3. If the data is correct, the user is redirected to their profile page.
4. If the data is incorrect, an error message is displayed, and the user repeats the process from step 1.

**US\_02: View personal record.**

Steps:

1. The user presses the “Record” section in the navigation bar.
2. Personal record is shown in the screen.

**US\_03: View list of doctors.**

Steps:

1. Users log in with the steps in US\_01.
2. The system redirects the user to the main page.
3. The list of all doctors is shown in the screen.

**US\_04: Edit personal information.**

Steps:

1. User navigates in personal record as in US\_02.
2. User clicks the edit button.
3. The user updates the required fields and presses the update button.
4. The system saves the new details.

**US\_05: Ask AI for recommendation**

Steps:

1. Users log in as patient with the steps in US\_01.
2. User writes the issue he has in the AI section.
3. List of doctors of the recommended department are shown.

**US\_06: Create a medical appointment**.

Steps:

1. User selects one of the doctors shown based on US\_03 to book an appointment.
2. User selects an available date, time and needed services.
3. User presses the book appointment button.
4. The system processes the booking and updates the time slots.

**US\_07: Download patient medical record**

Steps:

1. Doctor logs in.
2. Doctor navigates to the appointments section
3. Doctor selects a date to check the appointments
4. The appointment is selected and information is shown.
5. Doctor clicks the download pdf button
6. The medical record is downloaded in the device

**US\_8: Cancel a reservation**

Steps:

1. Users log in as doctor in US\_01.
2. The user navigates to the scheduled appointments section.
3. A list of upcoming appointments is shown.
4. Doctor views reservation details.
5. Doctor presses the cancellation button and the system is updated.

**US\_9: Mark appointment as complete.**

Steps:

1. Users log in as doctor in US\_01.
2. The user navigates to the scheduled appointments section.
3. A list of upcoming appointments is shown.
4. Doctor views reservation details.
5. Doctor presses the completion button and the system is updated.

**US\_10: View history of medical records.**

Steps:

1. Users logs in as patient with the steps in US\_01.
2. The user navigates to the history section in the nav bar, and button Records will be shown.
3. After clicking the Medical Record will appear.

**US\_11: View assigned patients list.**

Steps:

1. Users log in as doctor with the steps in US\_01.
2. The list of patients is directly shown in the screen by default.

**US\_12: View staff list.**

Steps:

1. Users log in as manager with the steps in US\_01.
2. The manager is redirected at the management dashboard.
3. Manager clicks the “Doctors” section in the nav bar.

**US\_13: Edit assigned patient medical record**

Steps:

1. The same steps as in US\_11.
2. The doctor clicks on a patient.
3. Doctor clicks on edit record button.
4. The doctor performs the edit then presses save changes.
5. Changes are saved and updated to be viewed by the system.

**US\_14: Add a staff member.**

Steps:

1. Manager logs in with the steps in US\_01.
2. The manager navigates to the “doctors” staff section.
3. Manager clicks the “Add doctor” button.
4. The modal is shown to fill the information for the new doctor.
5. Manager presses the confirmation button
6. Doctors list is updated with the new doctor.

**US\_15: Delete user or doctor**

Steps:

1. Manager views staff list as in US\_12.
2. Manager clicks the delete button in the right.
3. Modal for deletion is shown.
4. Manager clicks the delete button.
5. User or doctor is deleted from the system.

**US\_16: Change profile picture**

Steps:

1. Users log in with the steps in US\_01.
2. The user clicks the profile icon.
3. The user clicks on Change Profile Picture.
4. The user uploads the new profile picture.

**US\_17: Filter by department**

Steps:

1. Users log in with the steps in US\_01.
2. The user clicks on the Departments dropdown.
3. The user clicks on the desired Department.

**US\_18: Change password**

Steps:

1. Each role-user logs in with the steps in US\_01.
2. He/she clicks on the profile icon.
3. The user types the new password.
4. User clicks “Update Password”.
5. The password is updated by the system.

**US\_19: Assign personal doctor**

Steps:

1. Managers log in with the steps in US\_01.
2. The manager navigates to the Unassigned tab.
3. The manager selects an unassigned user.
4. The manager selects a doctor for the user.

**US\_20: Logout of the system**

Steps:

1. Users log in with the steps in US\_01.
2. The user navigates to the profile icon.
3. The user clicks the log out button.
4. The user is logged out of the system securely.

### **4.1.2 Use Case Tables**

UC 01: User logs in

|  |  |
| --- | --- |
| UC Name | User logs in the system |
| Summary | Allows users to log into the system using their credentials. |
| Dependency | None |
| Actors | Patient, Manager, Doctor |
| Preconditions | The user must have a registered account. |
| Description of the Main Sequence | - Step 1: User enters username and password.  - Step 2: User clicks login button.  - Step 3: System authenticates user.  -Step 4 : System identifies user role.  - Step 5: System redirects to role-specific dashboard |
| Description of the Alternative Sequence | 1.**Invalid Credentials :**  a. System displays error message ;  b. User re-enters credentials  2. **User registers a new account.** |
| Non functional requirements | The system should process login requests within 3 seconds. |
| Postconditions | The user is either logged in or prompted to retry. |

UC 02: View personal record

|  |  |
| --- | --- |
| UC Name | View personal record. |
| Summary | Patient view his own record, containing personal and health information |
| Dependency | UC 01 |
| Actors | Patient |
| Preconditions | User must be logged in |
| Description of the Main Sequence | - Step 1: Patient is logged in to the system and redirected to his page.  - Step 2: Patient clicks the Record section in the navigation bar.  - Step 3: The record containing the patients information is shown in the screen. |
| Description of the Alternative Sequence | None because a registered user always has a personal record. |
| Non functional requirements | Medical record should be shown within two seconds of button click. |
| Postconditions | The information of the record is shown in the screen |

UC 03: View list of doctors

|  |  |
| --- | --- |
| UC Name | View profile information of doctor. |
| Summary | Allows users to see the details about a doctor in order to categorize his/her specialization |
| Dependency | UC 01 |
| Actors | Patient, Manager, |
| Preconditions | User must have an existing profile. |
| Description of the Main Sequence | - Step 1: User logs in the system.  - Step 2: System retrieves the user to the main page.  -Step 3: User clicks the section “Doctors”  -Step 4: The list of all doctors is shown in the screen. |
| Description of the Alternative Sequence | None because a registered user has always option to show the doctors |
| Non functional requirements | Profile should load within 2 seconds. |
| Postconditions | The information with doctors is shown. |

UC 04: Edit personal information

|  |  |
| --- | --- |
| UC Name | Update personal information |
| Summary | Enables users to edit their personal details for any change they want to make. |
| Dependency | UC 01, UC 02 |
| Actors | Patient |
| Preconditions | User must have an existing profile. |
| Description of the Main Sequence | - Step 1: Patient is logged in to the system and redirected to his page.  - Step 2: Patient clicks the Record section in the navigation bar.  -Step 3: The information from the record will be shown.  -Step 4: Click button “Edit” in order to make changes.  -Step 5: After changes save new details. |
| Description of the Alternative Sequence | -Press “Cancel” button :   * Nothing is changed. |
| Non-functional requirements | Changes should reflect in the system within 2 seconds. |
| Postconditions | User profile information will be updated with new changes. |

UC 05: Ask AI for recommendation.

|  |  |
| --- | --- |
| UC Name | Ask AI for recommendation. |
| Summary | Is used AI to describe issue and it will recommend the right department. |
| Dependency | UC 01 |
| Actors | Patient |
| Preconditions | User must be logged in. |
| Description of the Main Sequence | -Step 1: User logs in system as patient  -Step 2: Will be shown in navbar the AI section, where he/she can describe issue.  -Step 3: The recommended departments from the AI will be shown |
| Description of the Alternative Sequence | None because a logged patient has AI section |
| Non-functional requirements | Profile should load within 2 seconds. |
| Postconditions | Based on issues, the right department will be shown. |

UC 06: Create a medical appointment in a free time-slot

|  |  |
| --- | --- |
| UC Name | Create medical appointment in a free time slot |
| Summary | Patients can select a slot which is available based on given date to book an appointment. |
| Dependency | UC 01 |
| Actors | Patient |
| Preconditions | There must be available time slots. |
| Description of the Main Sequence | -Step 1: User logs in system as patient.  -Step 2: Select a doctor to make an appointment  -Step 3: Select a free time slot in a day.  Step  -Step 4: Confirm the booking for next appointment, clicking the button “Booking”. |
| Description of the Alternative Sequence | - Step 1: No slots are available.  - Step 2: System provides an option to waitlist. |
| Non-functional requirements | Build an efficient system to perform fast operations. |
| Postconditions | Appointment is successfully booked. |

UC 07: Download patient medical record.

|  |  |
| --- | --- |
| UC Name | Download patient medical record |
| Summary | In order to have as offline file the doctor can download the medical patient record. |
| Dependency | UC 01 |
| Actors | Doctor |
| Preconditions | User must be log in. |
| Description of the Main Sequence | -Step 1: Doctor logs in as UC 01.  -Step 2: Doctor navigates to the appointments section  -Step 3: Doctor selects a date to check the appointments  -Step 4:The appointment is selected and information is shown.  -Step 5: Doctor clicks the download pdf button  -Step 6:The medical record is downloaded in the device |
| Description of the Alternative Sequence | - If the patient has no medical record available, no medical record will display |
| Non-functional requirements | Can be downloaded in 2 seconds |
| Postconditions | The medical record successfully downloaded. |

UC 08: Cancel a reservation

|  |  |
| --- | --- |
| UC Name | Cancel a reservation |
| Summary | Allows the doctors to cancel the booking for a different reason |
| Dependency | UC 01 |
| Actors | Doctor |
| Preconditions | The user must be logged as doctors in to access the reservations page. |
| Description of the Main Sequence | -Step 1: Users logs in system as doctor.  -Step 2: The page will be redirected and Appointments button will appear.  -Step 3: After clicking it, the list of “My Appointments” is shown.  -Step 4: For any patient you want to cancel booking, just click the profile.  -Step 5: Click the button “ Cancel Appointment”. |
| Description of the Alternative Sequence | None because if don’t have free slots the booking will be cancelled. |
| Non-functional requirements | The system must load services within 2 seconds, ensure mobile compatibility, and support text-to-speech for accessibility. |
| Postconditions | The patient can book a new appointment in another moment. |

UC 09: Mark appointment as completed.

|  |  |
| --- | --- |
| UC Name | Mark appointment as completed. |
| Summary | Allows doctors to complete the appointment after creating a new slot from the patient |
| Dependency | UC 01, UC 06 |
| Actors | Doctor |
| Preconditions | The user must be logged in to access the reservations page. |
| Description of the Main Sequence | -Step 1: User logs in as doctor in UC 01,  -Step 2: The user navigates all scheduled appointments  -Step 3: Doctor can view one of reservation detail.  -Step 4: He presses the completion button.  -Step 5: In patient profile the appointment will updated and marked as completed. |
| Description of the Alternative Sequence | The patient decides to change the appointment after is done completed. |
| Non-functional requirements | The system must load services within 2 seconds, ensure mobile compatibility, and support text-to-speech for accessibility. |
| Postconditions | The doctor accepts new appointment and it will be shown in “My patients” section. |

UC 10: View history of medical records.

|  |  |
| --- | --- |
| UC Name | View history of medical records. |
| Summary | Allows patient to see the list of medical records in order to use them for a health improvement |
| Dependency | UC 01 |
| Actors | Patient. |
| Preconditions | The user must be logged in as patient |
| Description of the Main Sequence | - Step 1: The user logs in system as patient.  -Step 2: The user navigates to history section in the nav bar and button “Records”.  -Step 3: The Medical Record will appear just to read and not to edit. |
| Description of the Alternative Sequence | If the patient had finished with treatment, no medical record will be shown. |
| Non-functional requirements | Cancellations should process within 3 seconds, confirmation should be sent via email, and user history must be updated accordingly. |
| Postconditions | The Medical Record appears with main details |

UC 11: View assigned patients list.

|  |  |
| --- | --- |
| UC Name | View assigned patients list |
| Summary | Enables doctors to see the list “My Patients” |
| Dependency | UC 01 |
| Actors | Doctor. |
| Preconditions | The user must be logged in. |
| Description of the Main Sequence | - Step 1: User logs in as doctor with steps in UC 01.  -Step 2: The lists of patients is directly shown in screen. |
| Description of the Alternative Sequence | If no patients exist, the lists will be empty. |
| Non-functional requirements | Efficient database access, secure data handling |
| Postconditions | The doctor will be available to see all patients that has. |

UC 12: View Staff List

|  |  |
| --- | --- |
| UC Name | View Staff List |
| Summary | Enables managers to see the staff list with all doctors in all departments. |
| Dependency | UC 01. |
| Actors | Manager |
| Preconditions | The user must be logged in |
| Description of the Main Sequence | -Step 1: Users log in as manager with the steps in US\_01.  -Step 2: The manager is redirected at the management dashboard.  -Step 3: Manager clicks the “Doctors” section in the nav bar. |
| Description of the Alternative Sequence | If list is empty, the system notifies the user |
| Non-functional requirements | Efficient database access, secure data handling. |
| Postconditions | The user successfully views the staff list. |

UC 13: Edit assigned patient medical record

|  |  |
| --- | --- |
| UC Name | Edit assigned patient medical record |
| Summary | The doctor can change the medical record based on health improvement. |
| Dependency | UC 01, UC 08. |
| Actors | Doctor |
| Preconditions | User must be logged in and have access rights. |
| Description of the Main Sequence | -Step 1: The  -Step 2: The doctor clicks on a patient. Doctor clicks on edit record button.  -Step 3: The doctor performs the edit then presses save changes.  -Step 4:Changes are saved and updated to be viewed by the system. |
| Description of the Alternative Sequence | If the doctor has not confirm as booking, no option for updating will be shown. |
| Non-functional requirements | Update should be sent via email, and user history must be updated accordingly. |
| Postconditions | The patients see medical record as updated. |

UC 14: Add a staff member.

|  |  |
| --- | --- |
| UC Name | Add a staff member. |
| Summary | The manager can add any member based on position or right department. |
| Dependency | UC 01. |
| Actors | Manager |
| Preconditions | User must be logged as manager in order to have availability to make changes on staff. |
| Description of the Main Sequence | -Step 1: Manager logs in with the steps in US\_01.  -Step 2: The manager navigates to the “doctors” staff section.  -Step 3: Manager clicks the “Add doctor” button.  -Step 4: The modal is shown to fill the information for the new doctor.  -Step 5: Manager presses the confirmation button  -Step 6: Doctors list is updated with the new doctor. |
| Description of the Alternative Sequence | If the department will be completed with staff, no option to add staff will appear |
| Non-functional requirements | Changes should reflect in the system within 2 seconds. |
| Postconditions | The changes will appear in staff list as updated. |

UC 5: Delete user or doctor .

|  |  |
| --- | --- |
| UC Name | Delete user or doctor. |
| Summary | The manager, except adding someone, have availability to delete. |
| Dependency | UC 01, UC 11 |
| Actors | Manager |
| Preconditions | User must be logged as manager in order to have availability to make changes on staff. |
| Description of the Main Sequence | -Step 1: Manager views staff list as in US\_12.  -Step 2: Manager clicks the delete button in the right.  -Step 3: Modal for deletion is shown.  -Step 4: Manager clicks the delete button.  -Step 5: User or doctor is deleted from the system. |
| Description of the Alternative Sequence | If anyone wants to decide to stay or department decide to keep, the deletion can be cancelled. |
| Non-functional requirements | Changes should reflect in the system within 2 seconds. |
| Postconditions | The changes should appear as updated. |

UC 16: Change profile photo.

|  |  |
| --- | --- |
| UC Name | Change profile photo. |
| Summary | The user after a long time can be update of profile and one details is the profile photo |
| Dependency | UC 01. |
| Actors | Manager |
| Preconditions | The user must be logged in |
| Description of the Main Sequence | -Step 1: Users log in with the steps in US\_01.  -Step 2: The user clicks the profile icon.  -Step 3: The user clicks on Change Profile Picture.  -Step 4:The user uploads the new profile picture. |
| Description of the Alternative Sequence | If anyone wants to decide to leave without photo, he can do it as well. |
| Non-functional requirements | Changes should reflect in the system within 2 seconds. |
| Postconditions | The new version of photo appeared. |

UC 17: Filter by department.

|  |  |
| --- | --- |
| UC Name | Filter by department. |
| Summary | The user can select just by using department filter, for the treatment they want. |
| Dependency | UC 01. |
| Actors | Patient |
| Preconditions | The user must be logged in. |
| Description of the Main Sequence | -Step 1: Users log in as UC 01.  -Step 2: The user clicks on the Department dropdown.  -Step 3: The user clicks on the desired Departament |
| Description of the Alternative Sequence | The patient goes to the list of Departments but does not choose through the filters, deciding to return to the beginning and receive AI recommandation. |
| Non-functional requirements | Changes should reflect in the system within 2 seconds. |
| Postconditions | One or many departments will appear based on filters. |

UC 18: Change password.

|  |  |
| --- | --- |
| UC Name | Change password. |
| Summary | Anyone can change the password of their account for maintenance or in cases where they forget the current password |
| Dependency | UC 01. |
| Actors | Patient, Doctor, Manager |
| Preconditions | The user must be logged in. |
| Description of the Main Sequence | -Step 1: Each role-user logs in with the steps in US\_01.  -Step 2: He/she clicks on the profile icon.  -Step 3: The user types the new password.  -Step 4: User clicks “Update Password”.  -Step 4: The password is updated by the system. |
| Description of the Alternative Sequence | The patient goes to the list of Departments but does not choose through the filters, deciding to return to the beginning and receive AI recommandation. |
| Non-functional requirements | Update should be sent via email, and user history must be updated accordingly |
| Postconditions | The new version of photo appeared. |

UC 19: Assign personal doctor.

|  |  |
| --- | --- |
| UC Name | Assign personal doctor. |
| Summary | Manager can assign a personal doctor for each patient, who is somehow more responsible for the continuation of his treatment. |
| Dependency | UC 01. |
| Actors | Manager |
| Preconditions | The user must be logged in. |
| Description of the Main Sequence | -Step 1: Managers log in with the steps in US\_01.  -Step 2: The manager navigates to the Unassigned tab.  -Step 3:The manager selects an unassigned user.  -Step 4: The manager selects a doctor for the user. |
| Description of the Alternative Sequence | If the list of unassigned is empty, back User Management section to add user. |
| Non-functional requirements | Update should be sent via email, and user history must be updated accordingly |
| Postconditions | The new patient will be assigned at personal doctor |

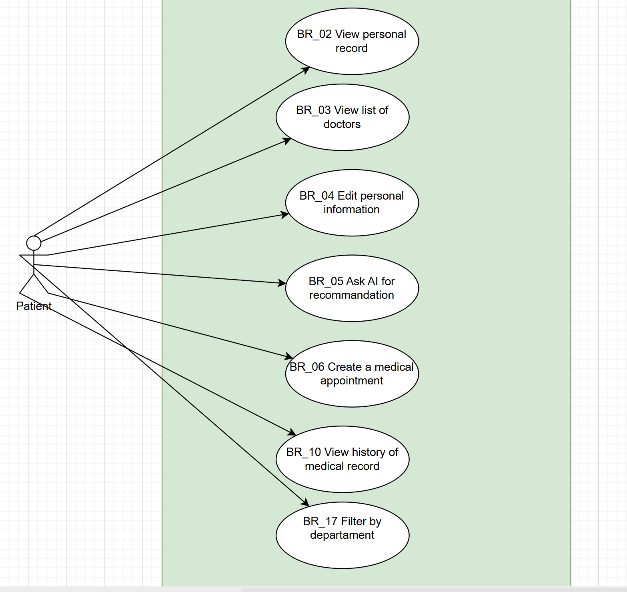
UC 20: Logout the system.

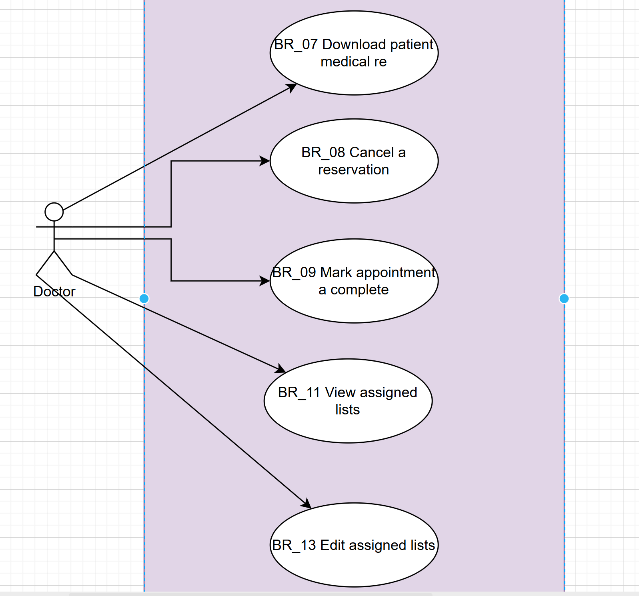
|  |  |
| --- | --- |
| UC Name | Assign personal doctor. |
| Summary | Anytime the user can leave the application doing sign out. |
| Dependency | UC 01. |
| Actors | Patient, Manager, Doctor. |
| Preconditions | The user must be logged in. |
| Description of the Main Sequence | -Step 1: Users log in with the steps in US\_01.  -Step 2: The user navigates to the profile icon.  -Step 3: The user clicks the log out button.  -Step 4: The user is logged out of the system securely. |
| Description of the Alternative Sequence | If the user want to save the pass for next time, a small window will appear to save it. |
| Non-functional requirements | Log out should be directly in 2 sec. |
| Postconditions | The user will sign out from app. |

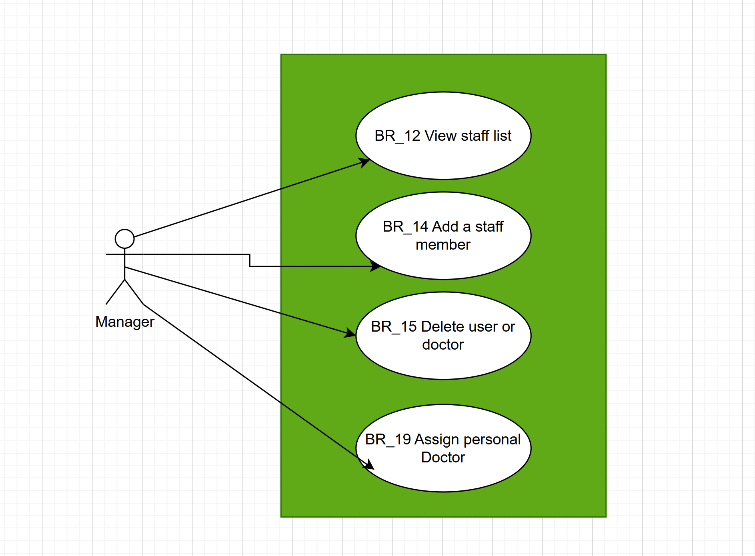
## **4.2** **Behavioral Diagrams**

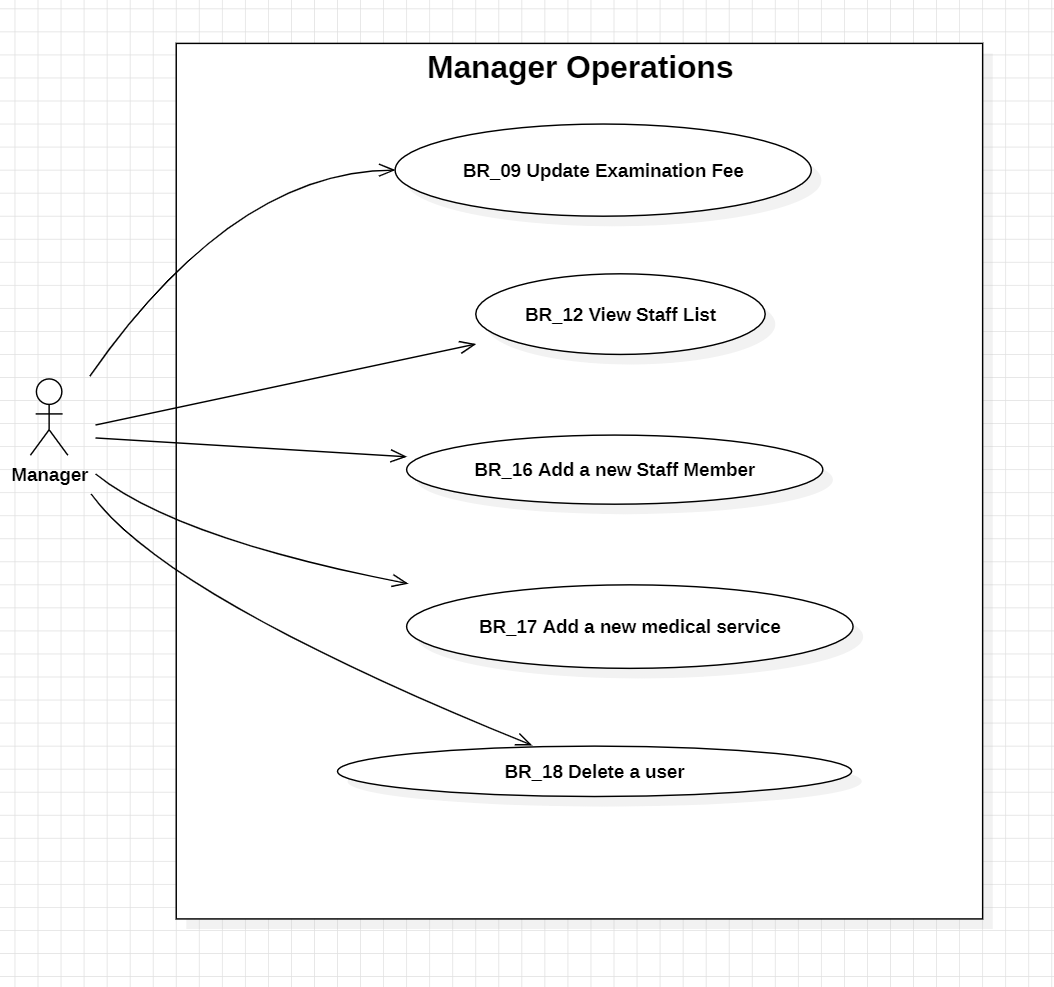
### **4.2.1 Use Case Diagrams**

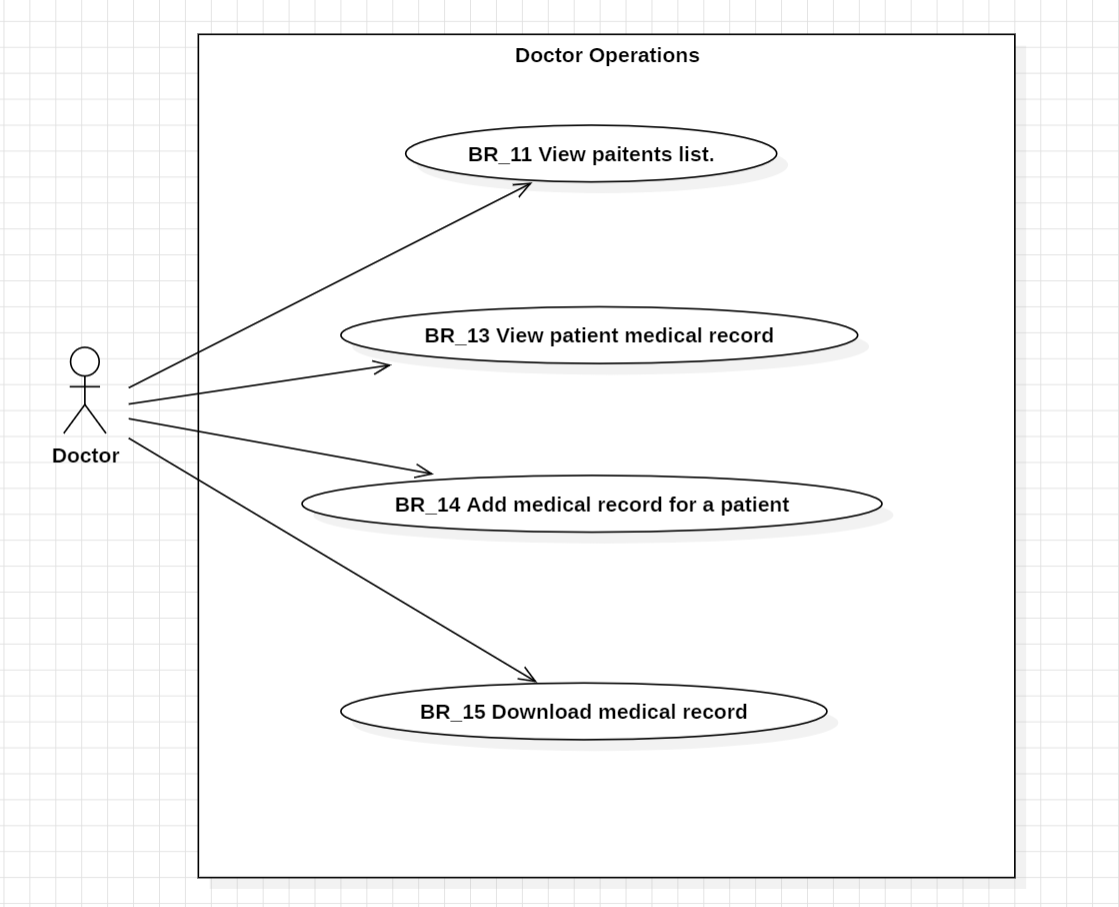


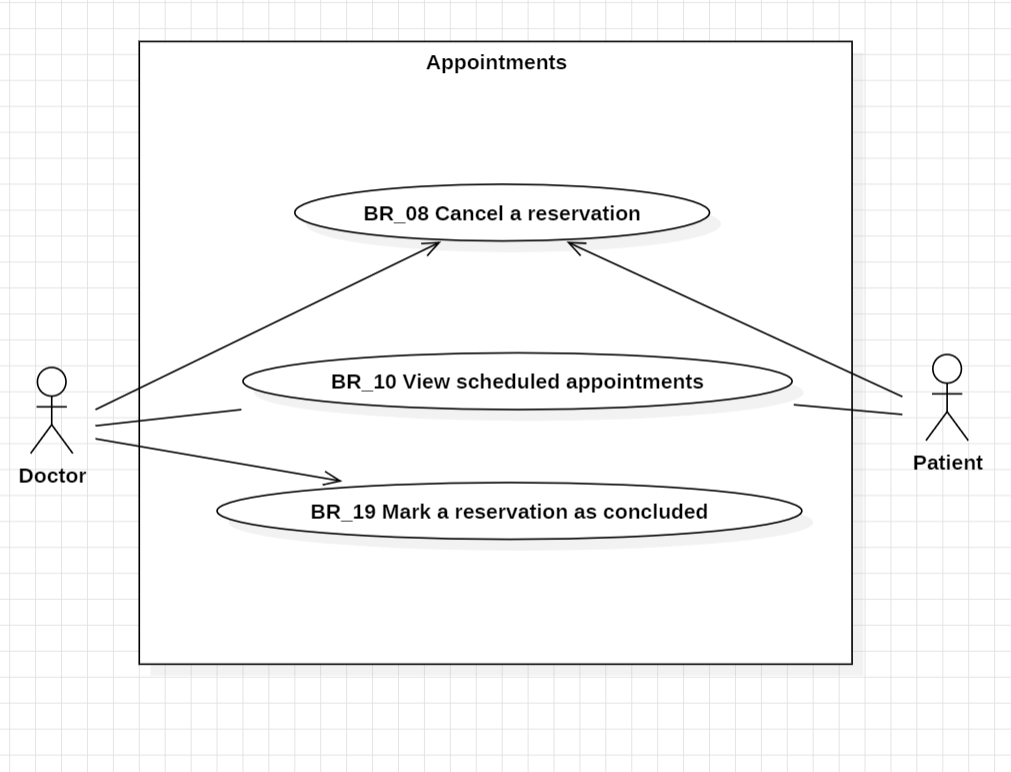










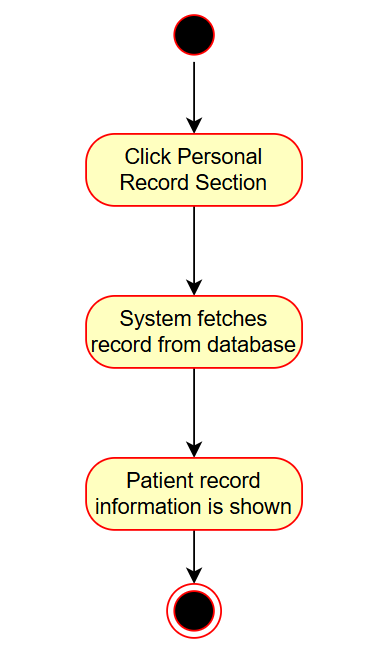


### **4.2.2 Activity Diagrams**

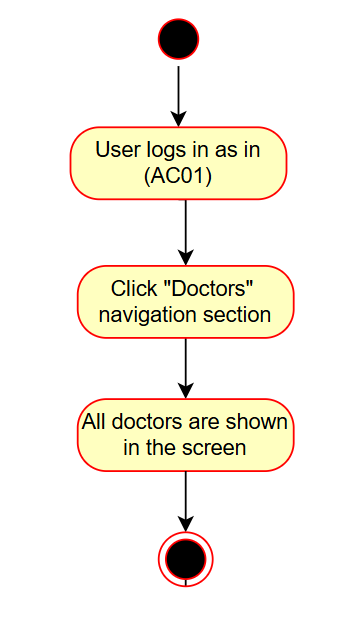
UC1



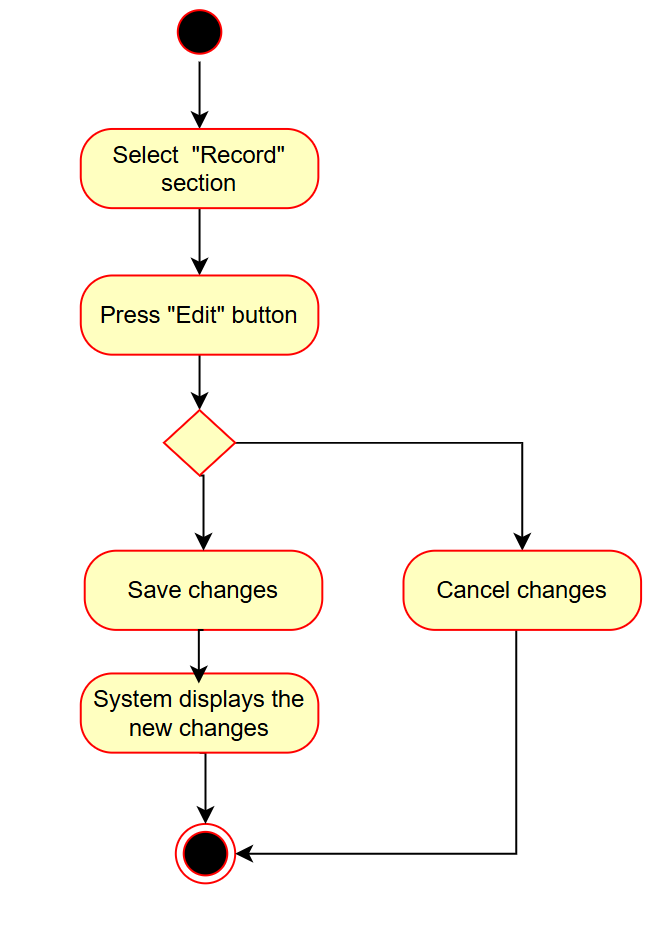
UC 2



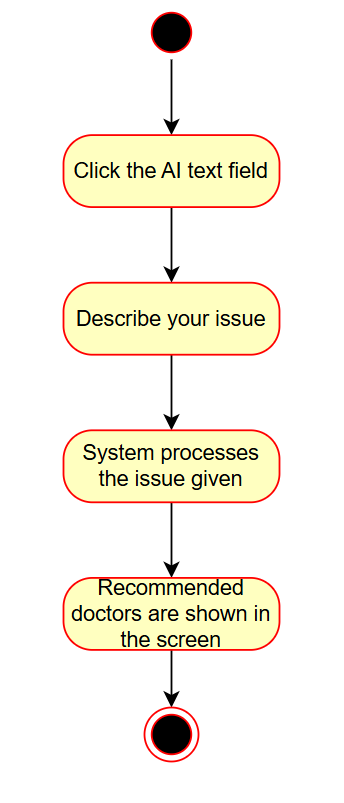
UC 3



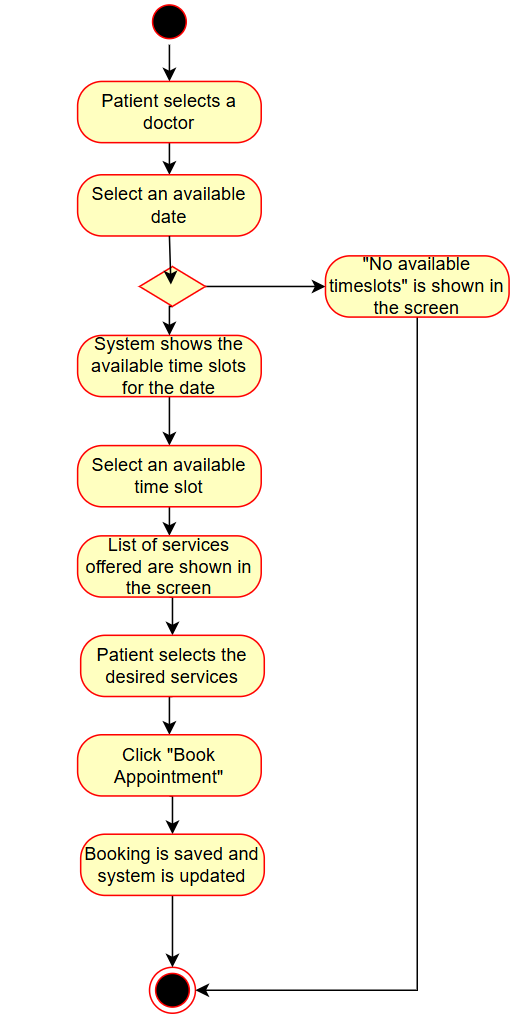
UC 4



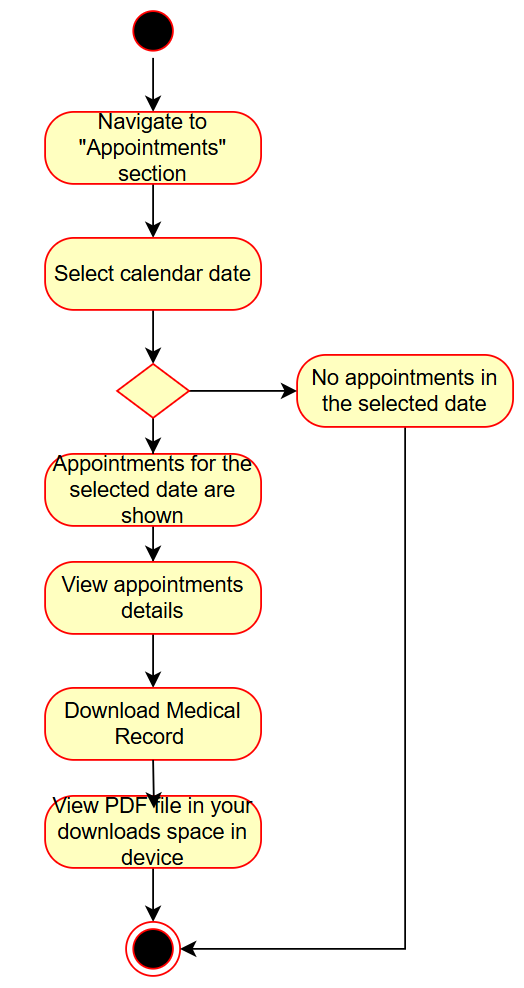
UC5



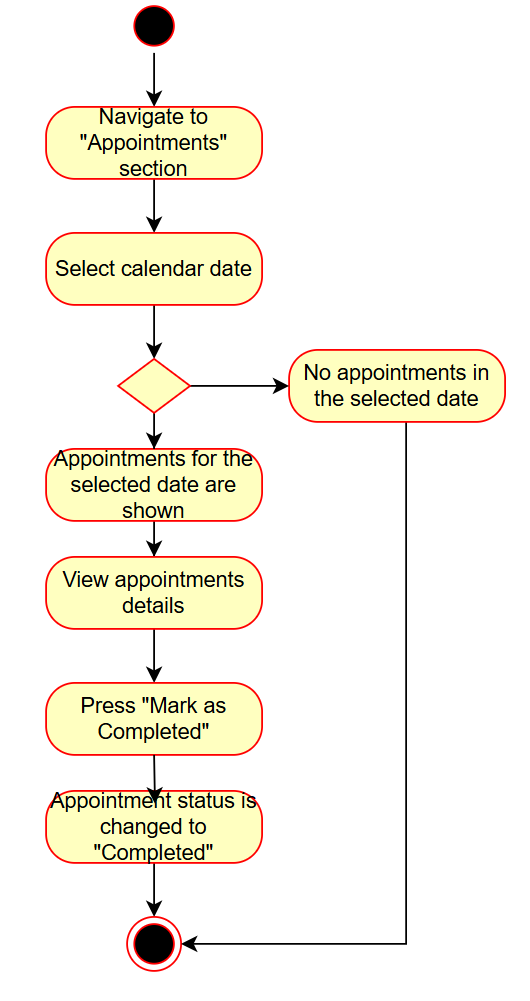
Uc6



UC 7

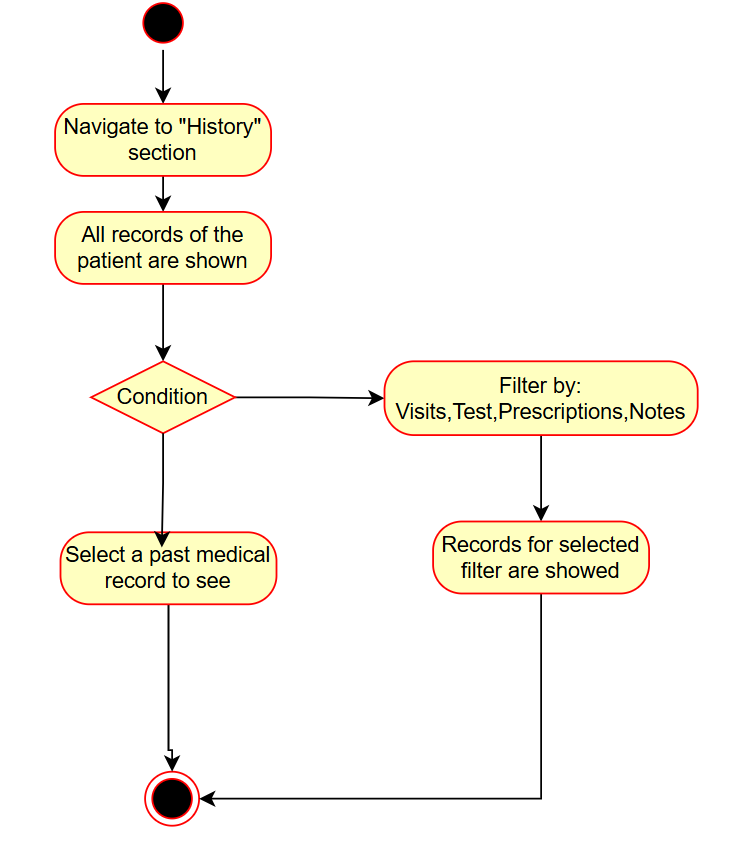


UC 8

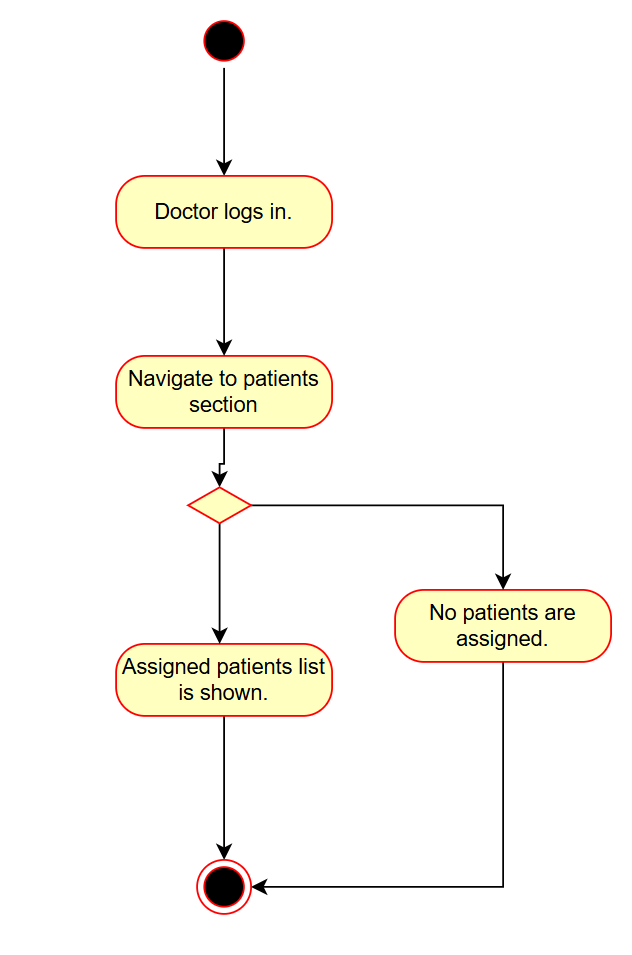


\

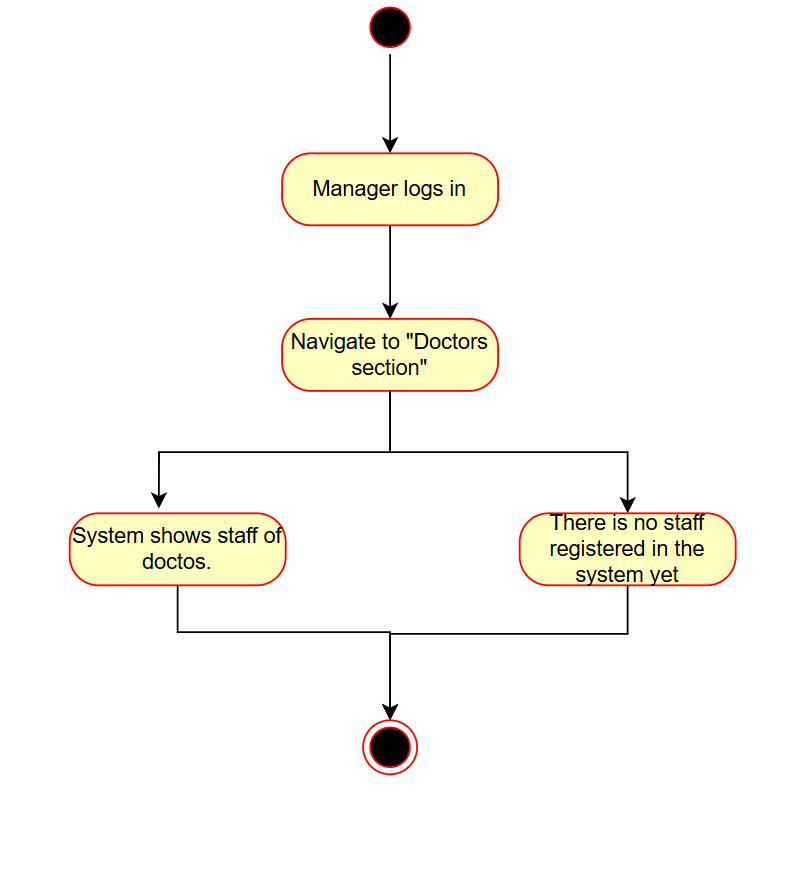
UC9



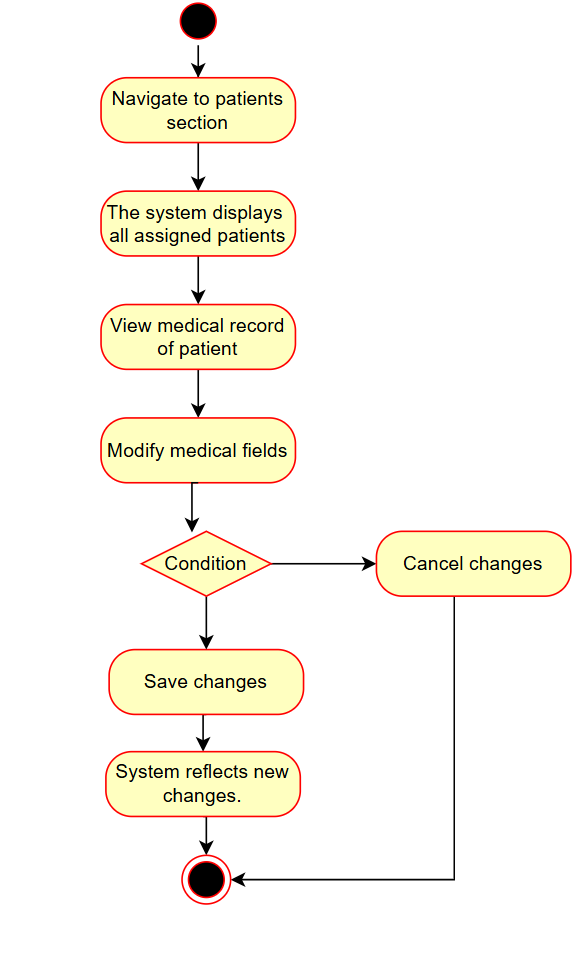
UC10



UC11



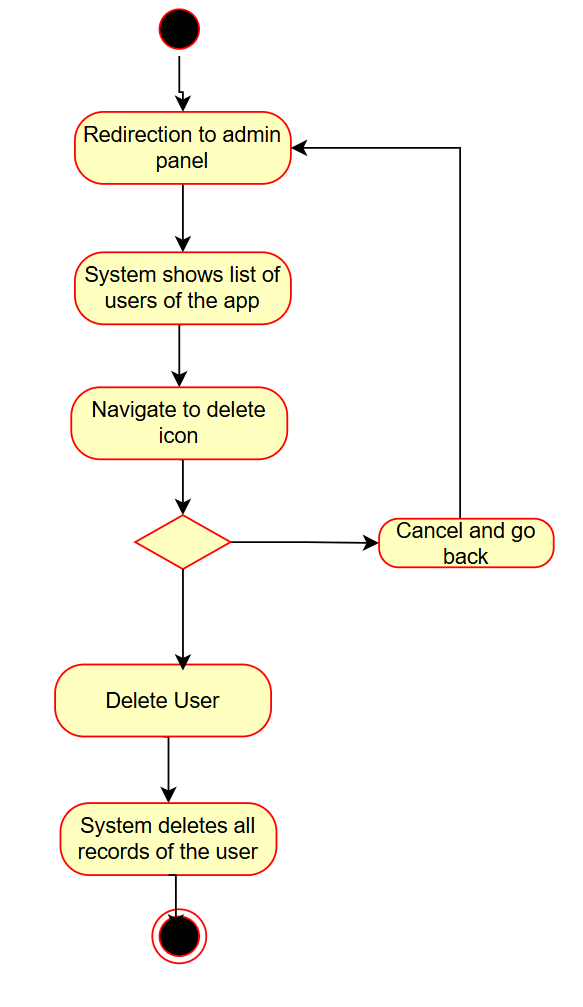
UC12



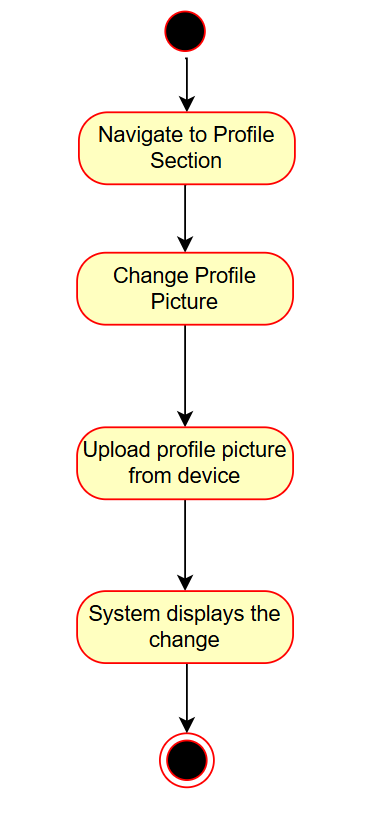
UC13



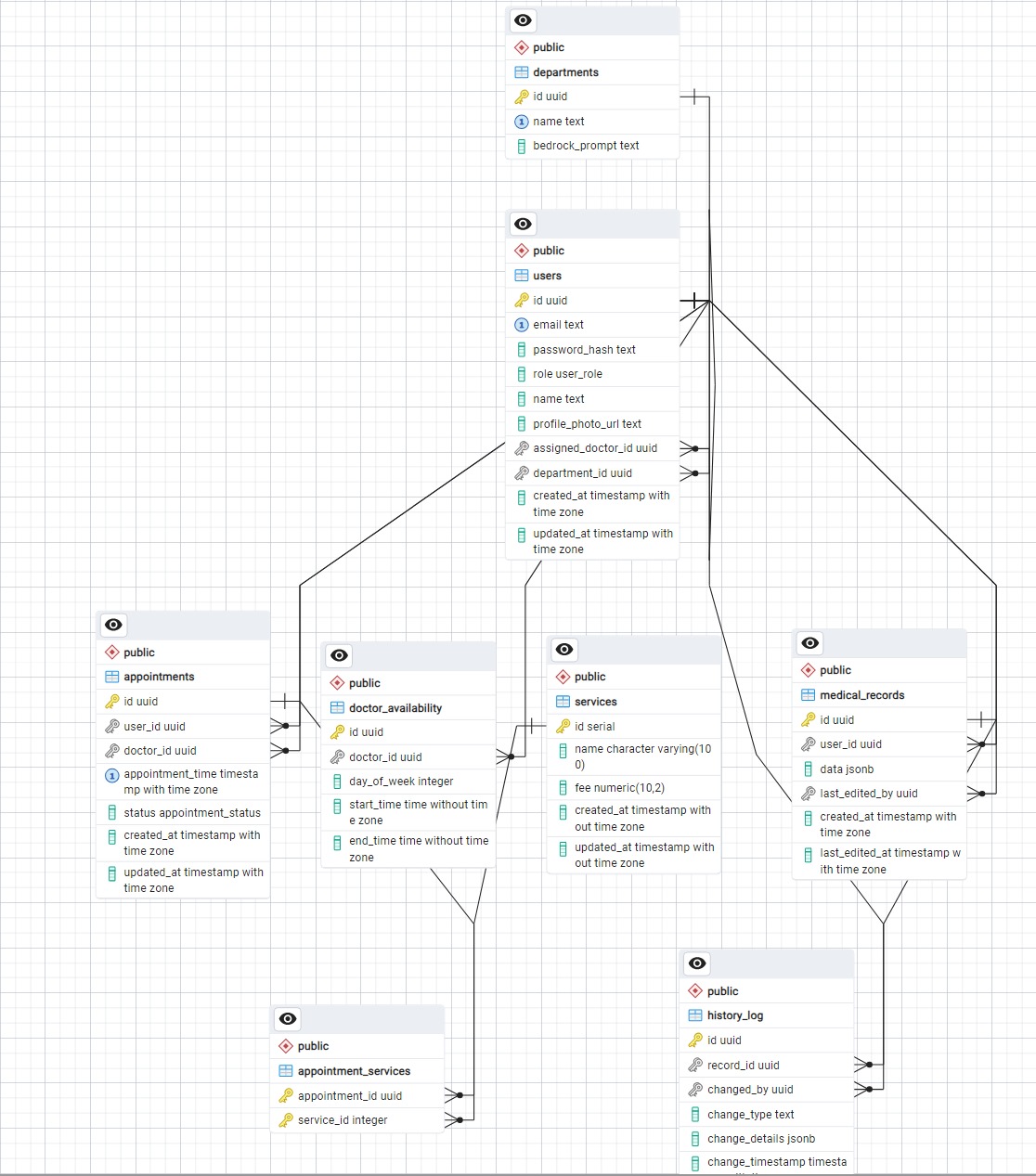
UC14



UC15



### **4.2.3 ER Diagram**



### **4.2.4 Class Diagrams**

Class relationships

A screenshot of a computer

AI-generated content may be incorrect.

Models class diagrams

A group of blue boxes with black text

AI-generated content may be incorrect.

Controller class diagrams

A screenshot of a computer program

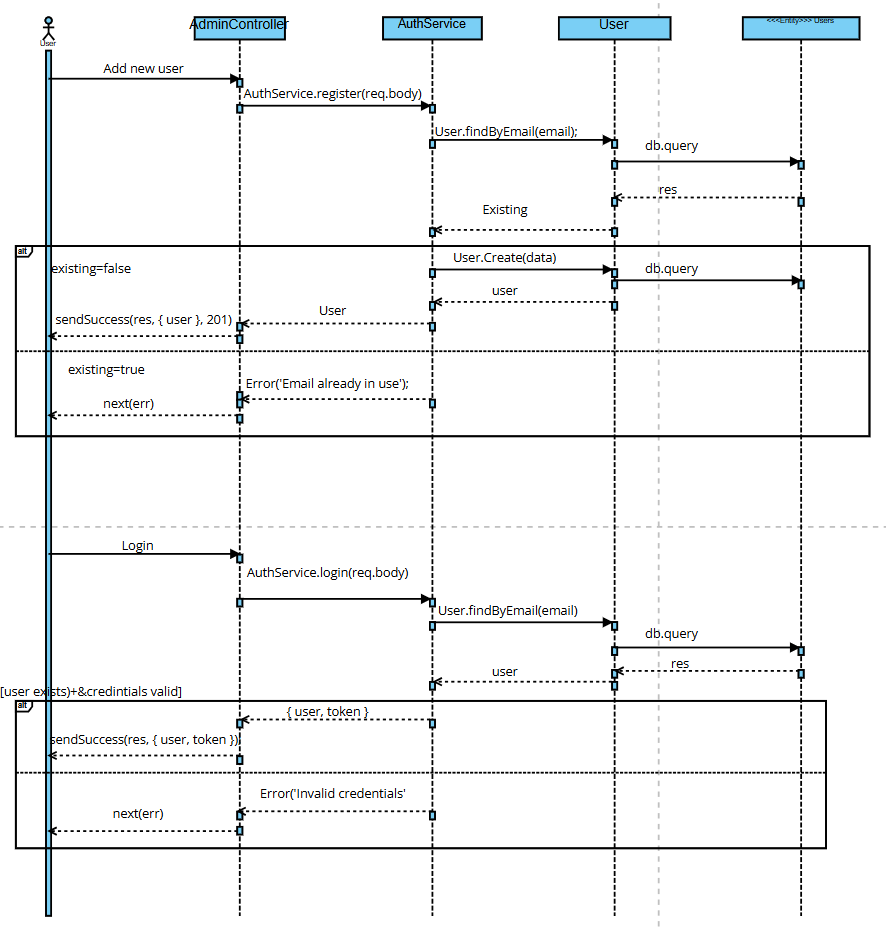
AI-generated content may be incorrect.

Services class diagrams

A screenshot of a computer program

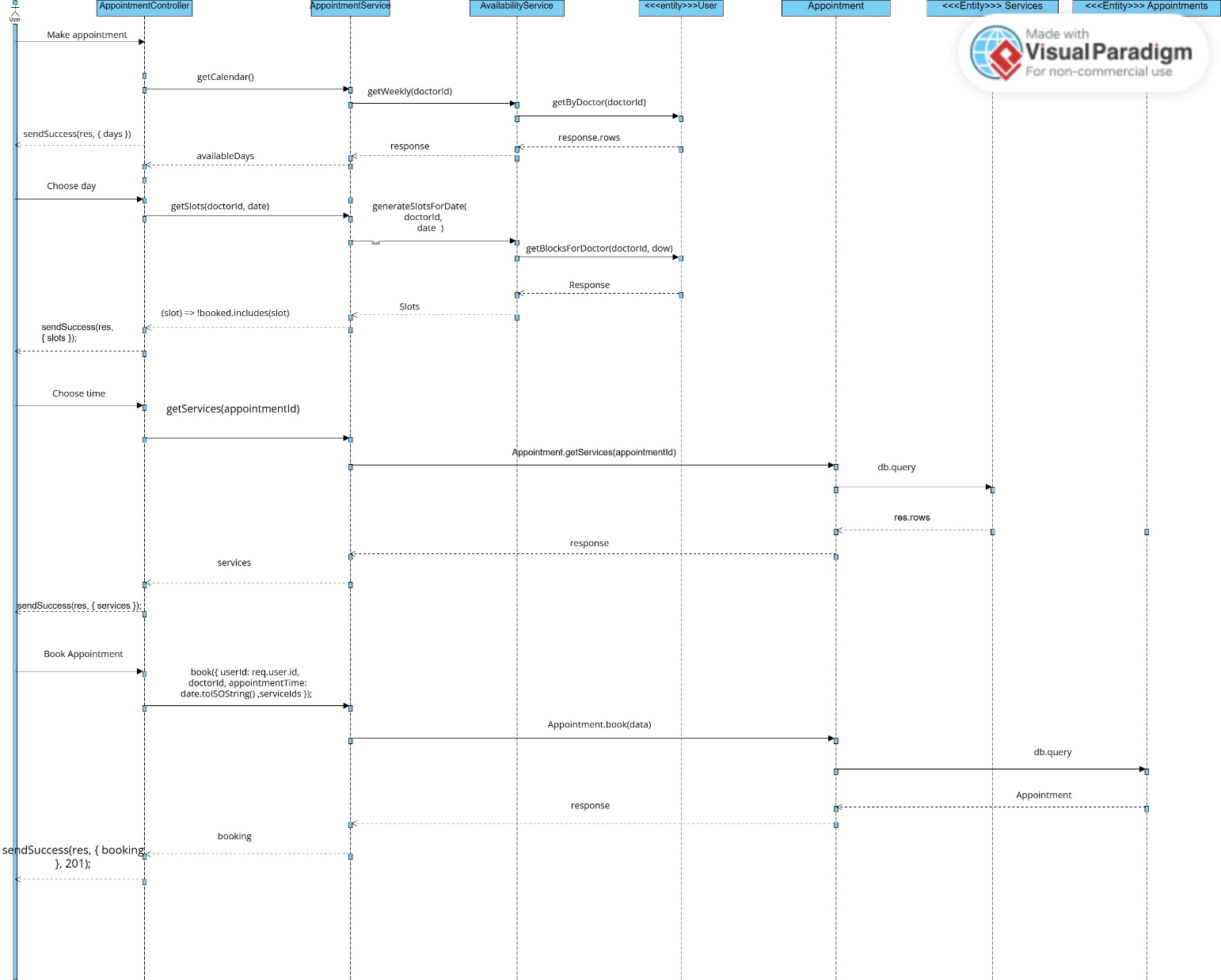
AI-generated content may be incorrect.

### **4.2.5 Sequence Diagrams**

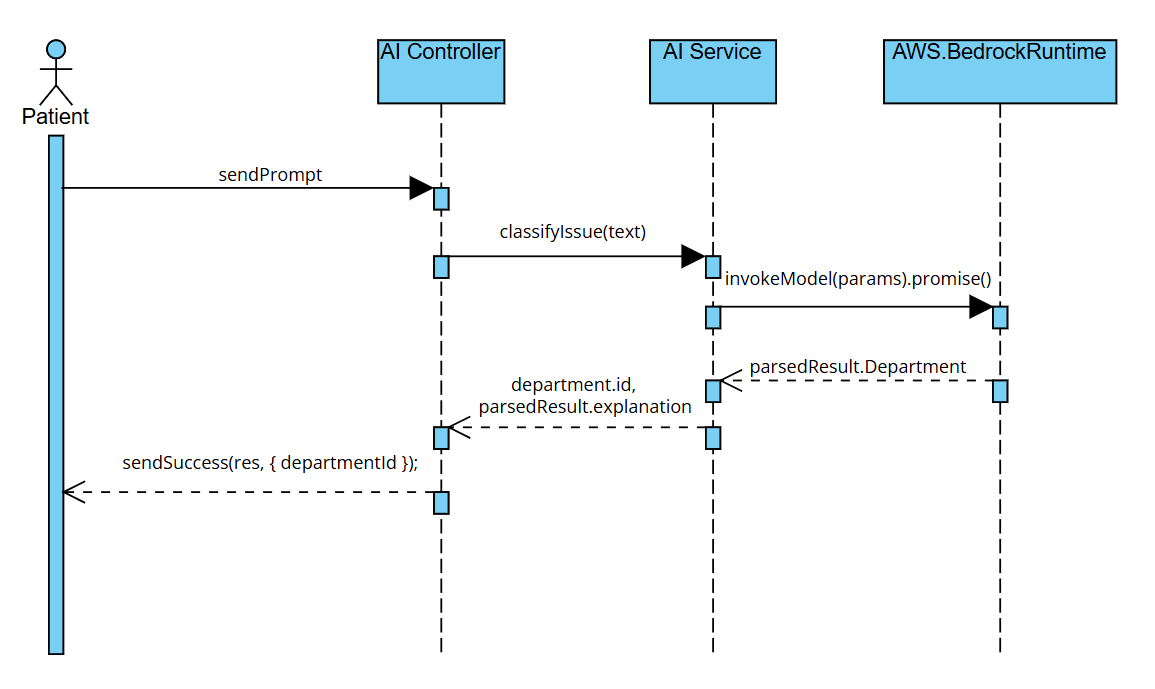


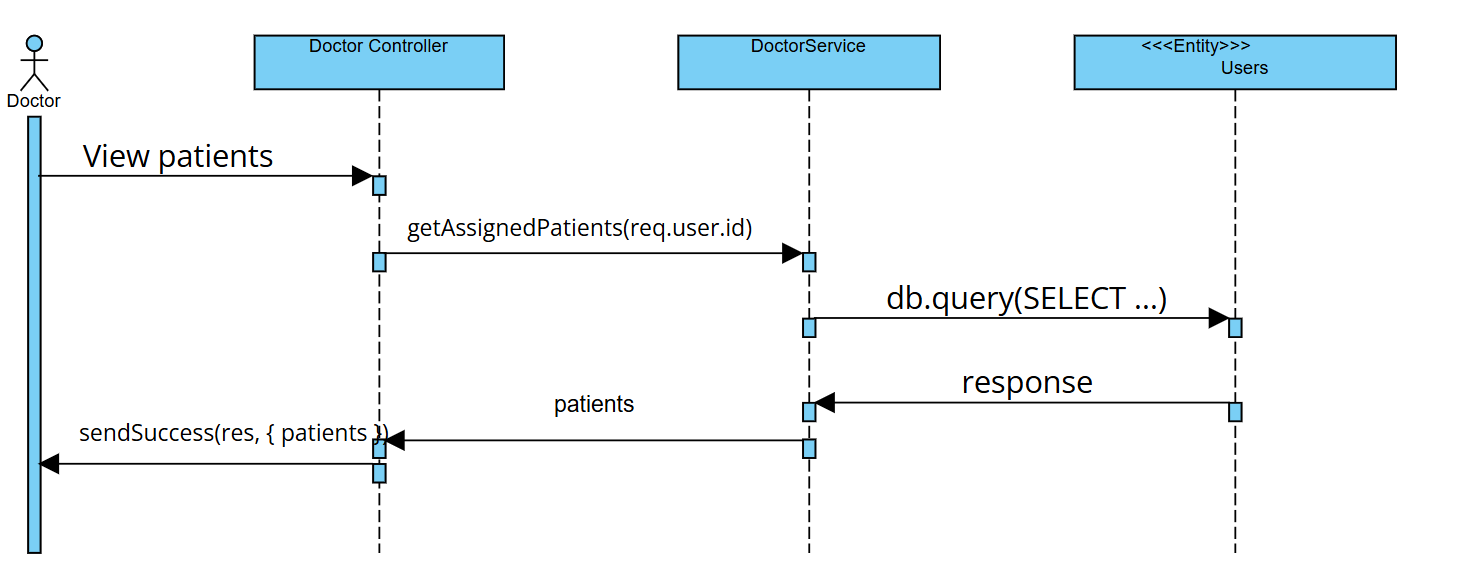
SeqD1 User logs in

UC 6

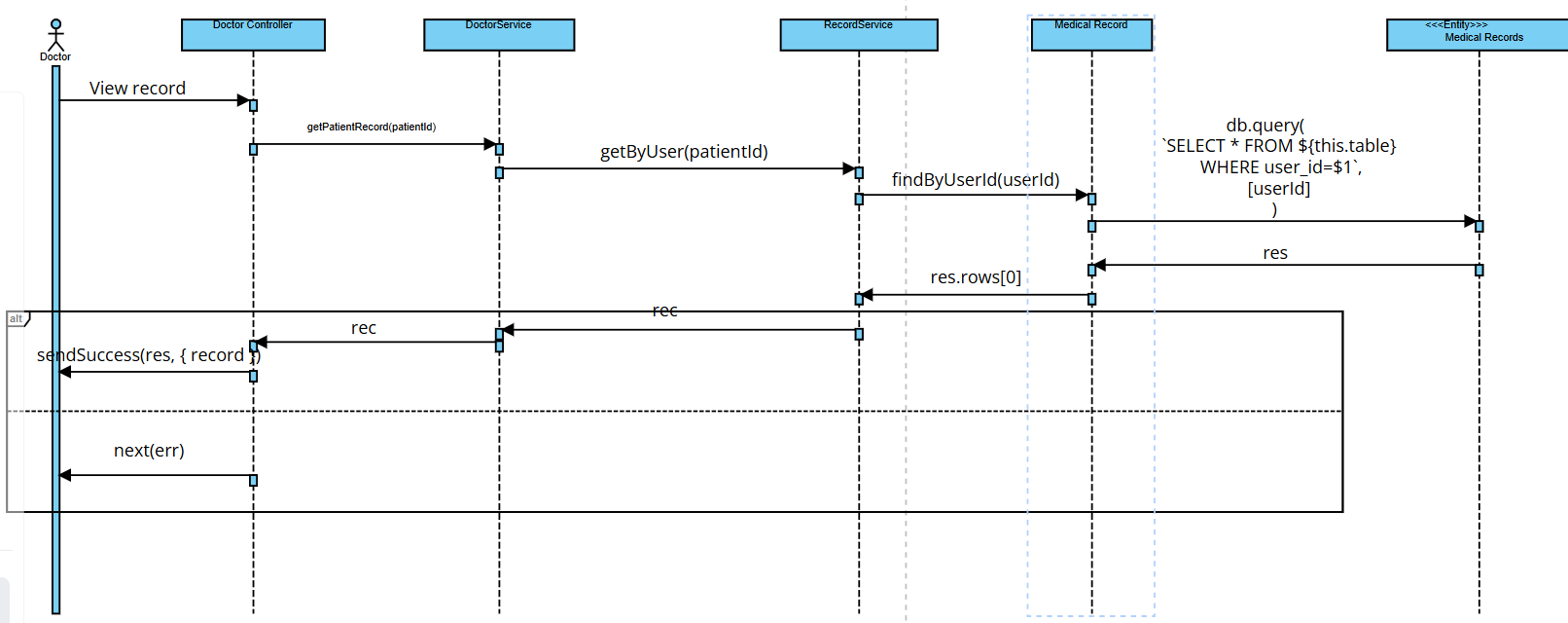


UC5 AI

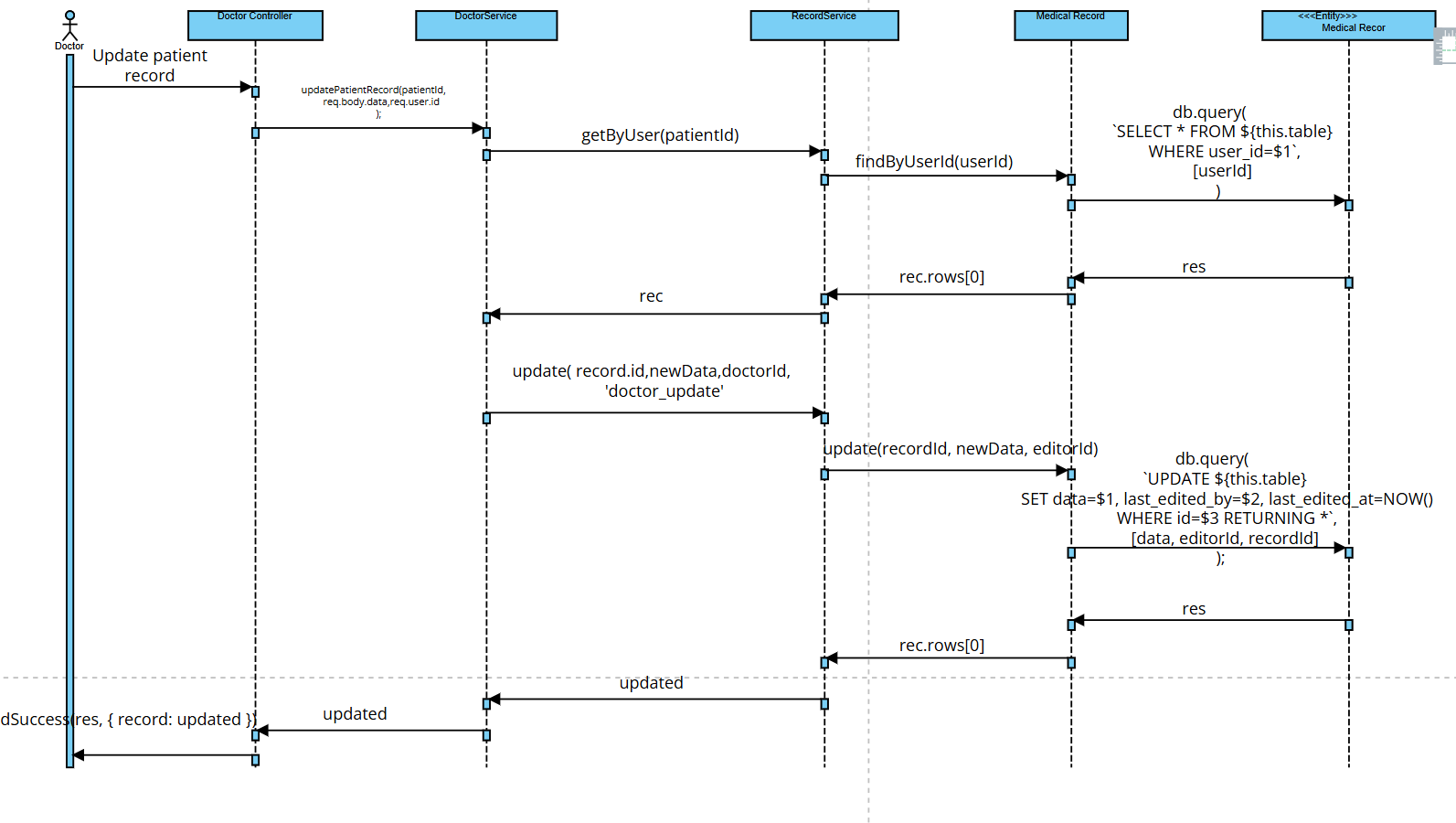




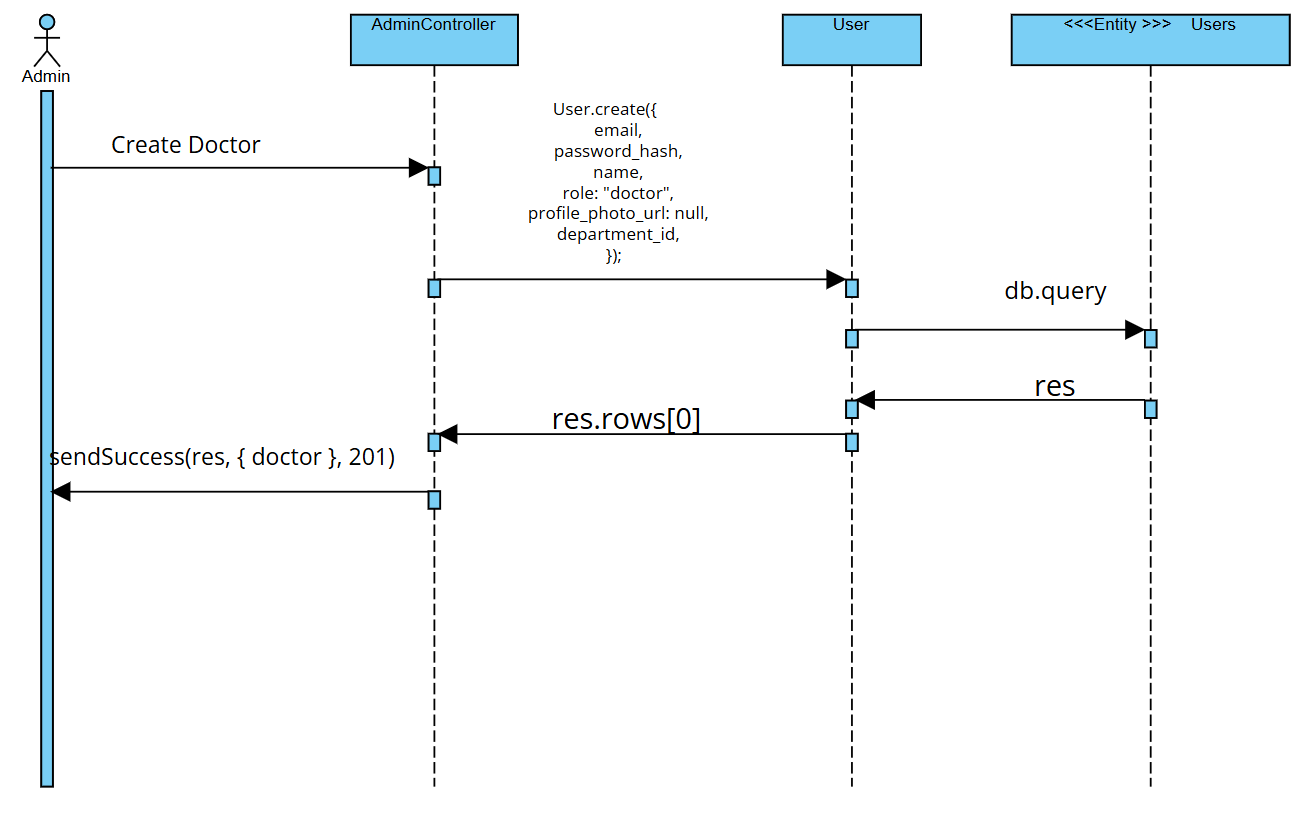
UC 11Doctor views its patients



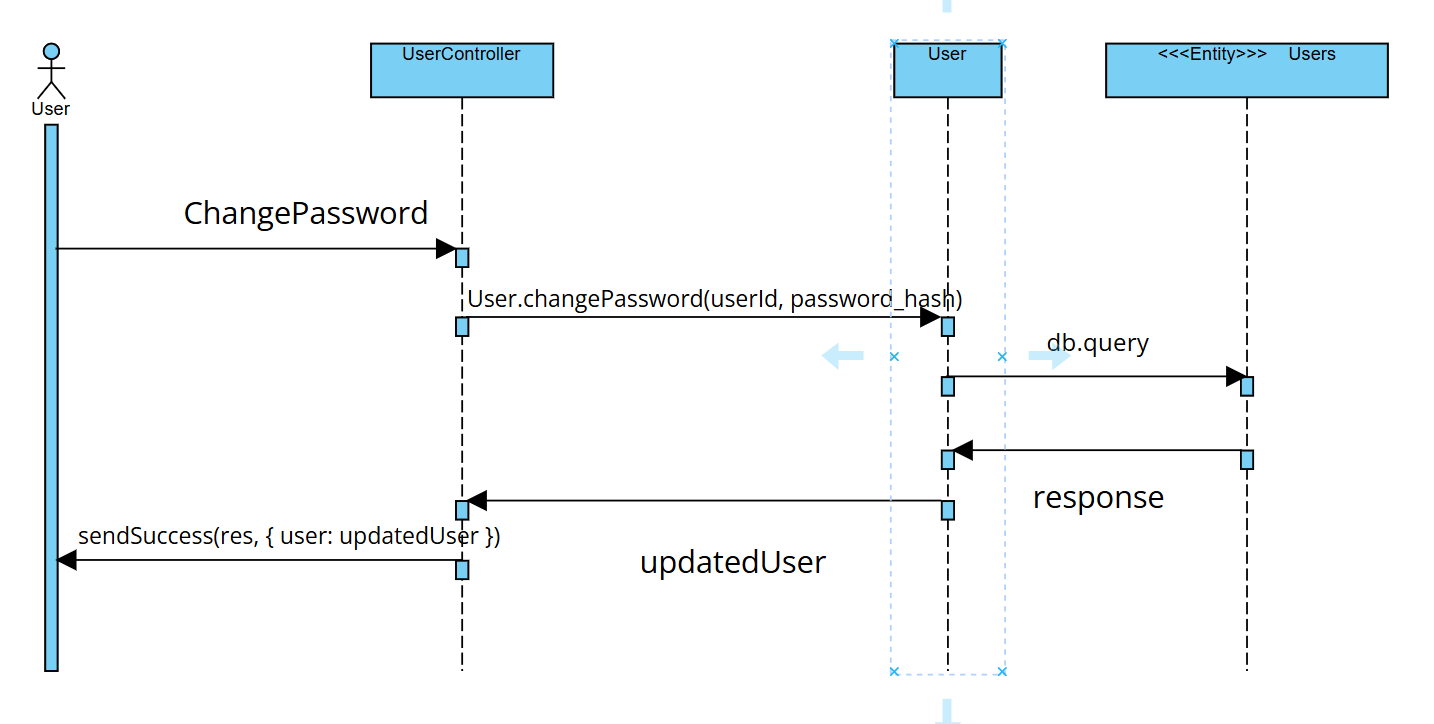
SeqD 9 View Medical Record



UC13 Update Medical Record

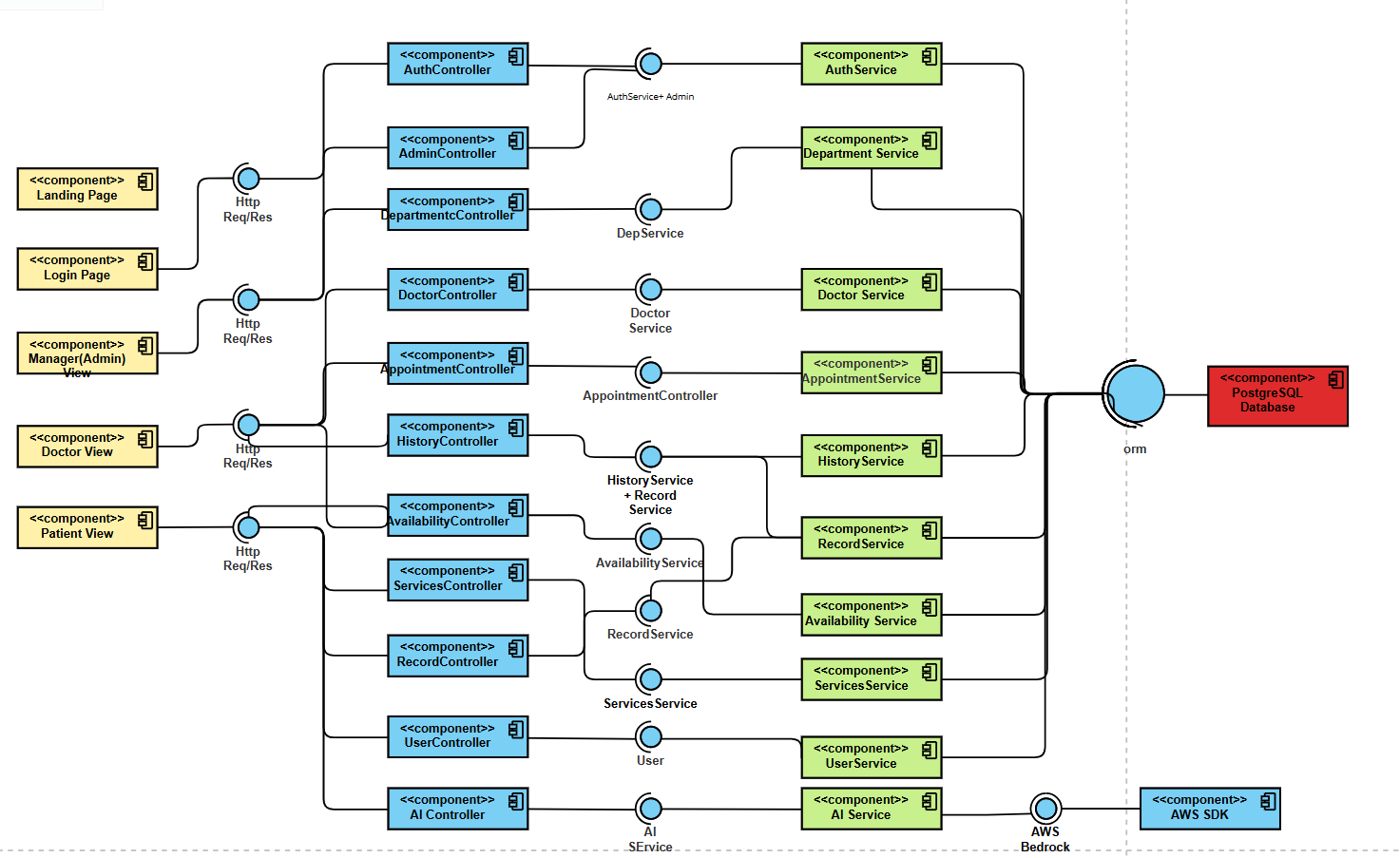


UC 14 Admin creates new doctor

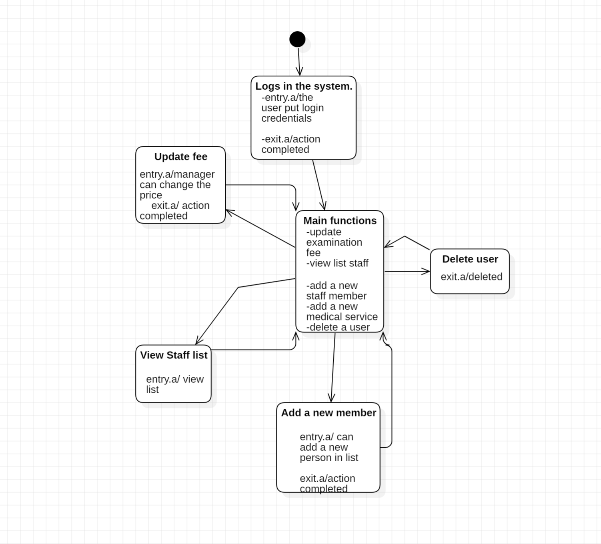


UC18 User changes password

### **4.2.6 Components Diagram**



### **4.2.7 State Diagram**



### **4.2.8 Project Planning**

Real start and end days: 9.03.2025-30.05.2025

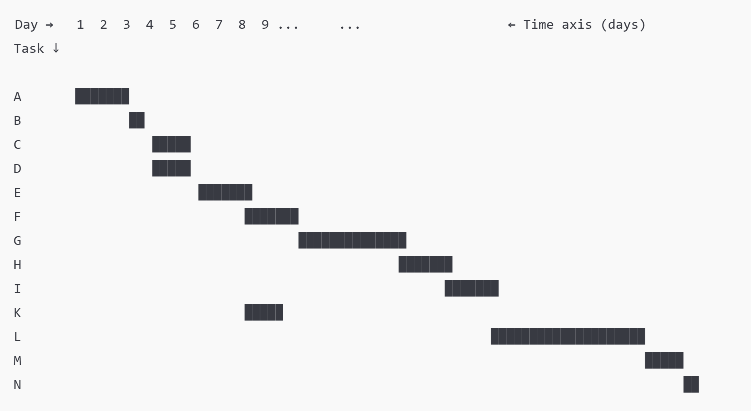
Estimated start and end days: 15.03.2025-15.06.2025

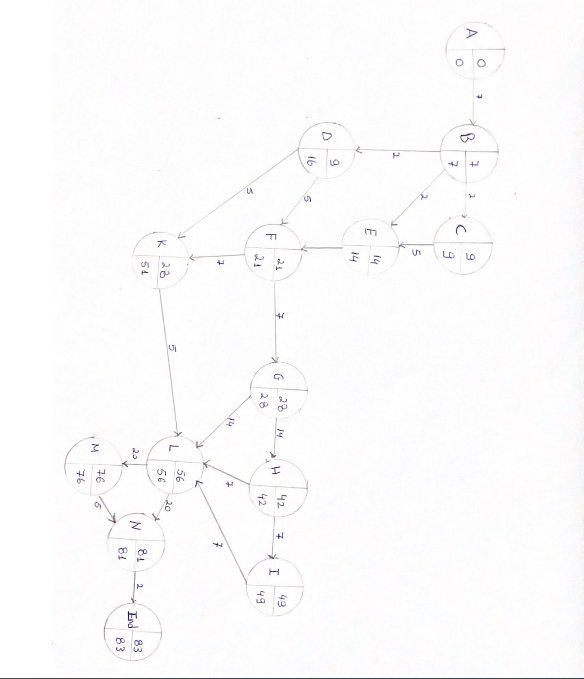
Real total days: 83 days

Estimated total days: 93

|  |  |  |  |
| --- | --- | --- | --- |
| Nr | Activity | Duration(Days) | Dependencies |
| A | Proposed topic for project | 7 |  |
| B | Feasibility Study | 2 | A |
| C | Project Description | 5 | B |
| D | Sketch Designs | 5 | B |
| E | Requirements (Functional,  Non-functional, Domain) | 7 | B, C |
| F | User scenarios  Use cases | 7 | D, E |
| G | Activity Diagram  State Diagrams | 14 | F |
| H | Sequence Diagrams | 7 | G |
| I | Class Diagram  Component Diagram | 7 | H |
| K | Detailed Design | 5 | D, F |
| L | Programming | 20 | G, H, I, K |
| M | Testing | 5 | L |
| N | Installation | 2 | L, M |

Gantt Chart





# **5. Technologies Used**

The project leverages a modern, full-stack JavaScript framework, known as the PERN stack:

- **Postgres:** Robust relational database management system for handling hospital data.

- **Express:** Fast and flexible Node.js web application framework for backend APIs.

**- React:** Modern, component-based library for building responsive and interactive user interfaces.

- **Node.js:** JavaScript runtime for executing server-side code, ensuring seamless integration with Express.

- **Deployment:** The application will be deployed on AWS, leveraging the AWS SDK v2 for JavaScript.

- AWS cloud deployment with EC2, RDS, and S3.

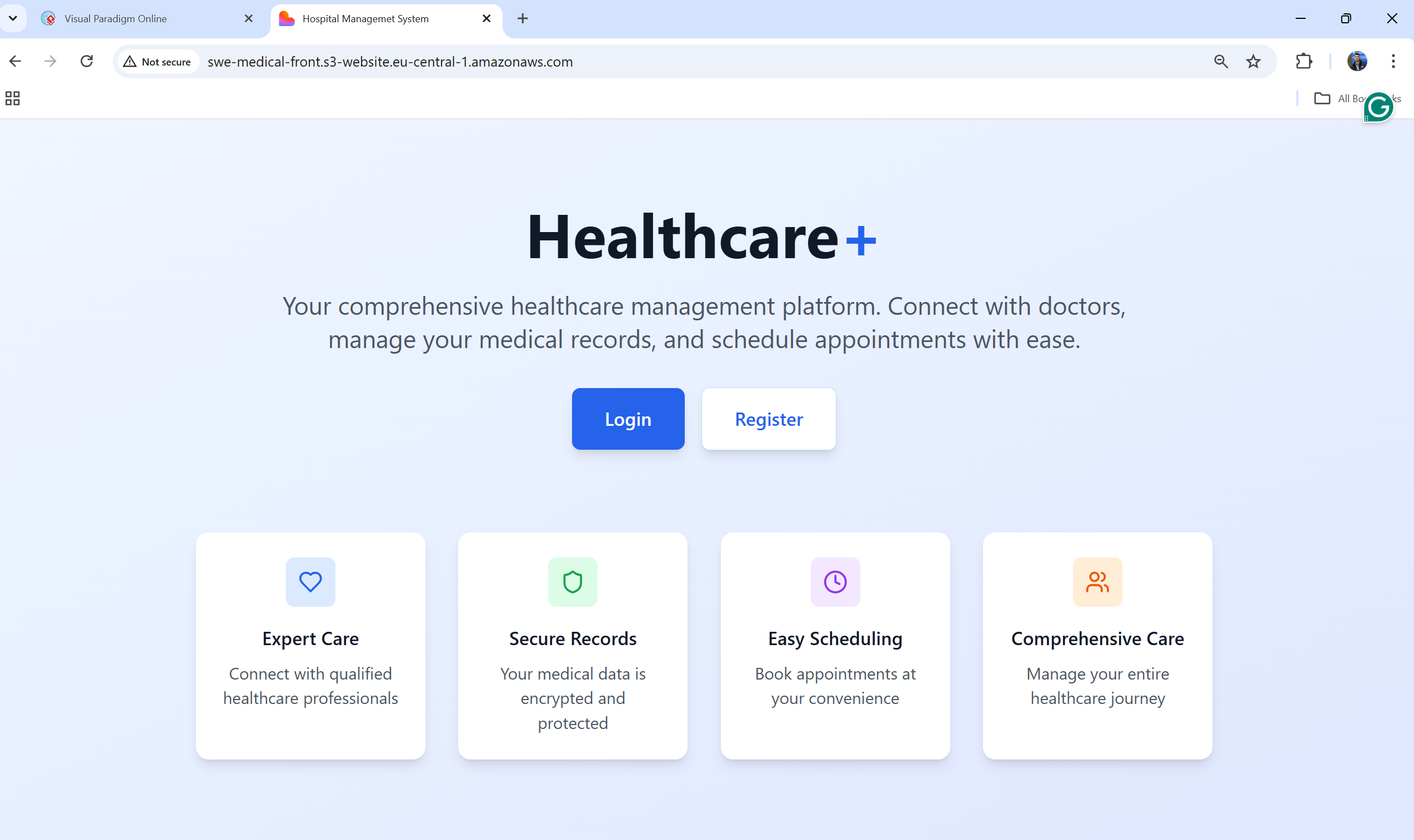
**Trello Board:**

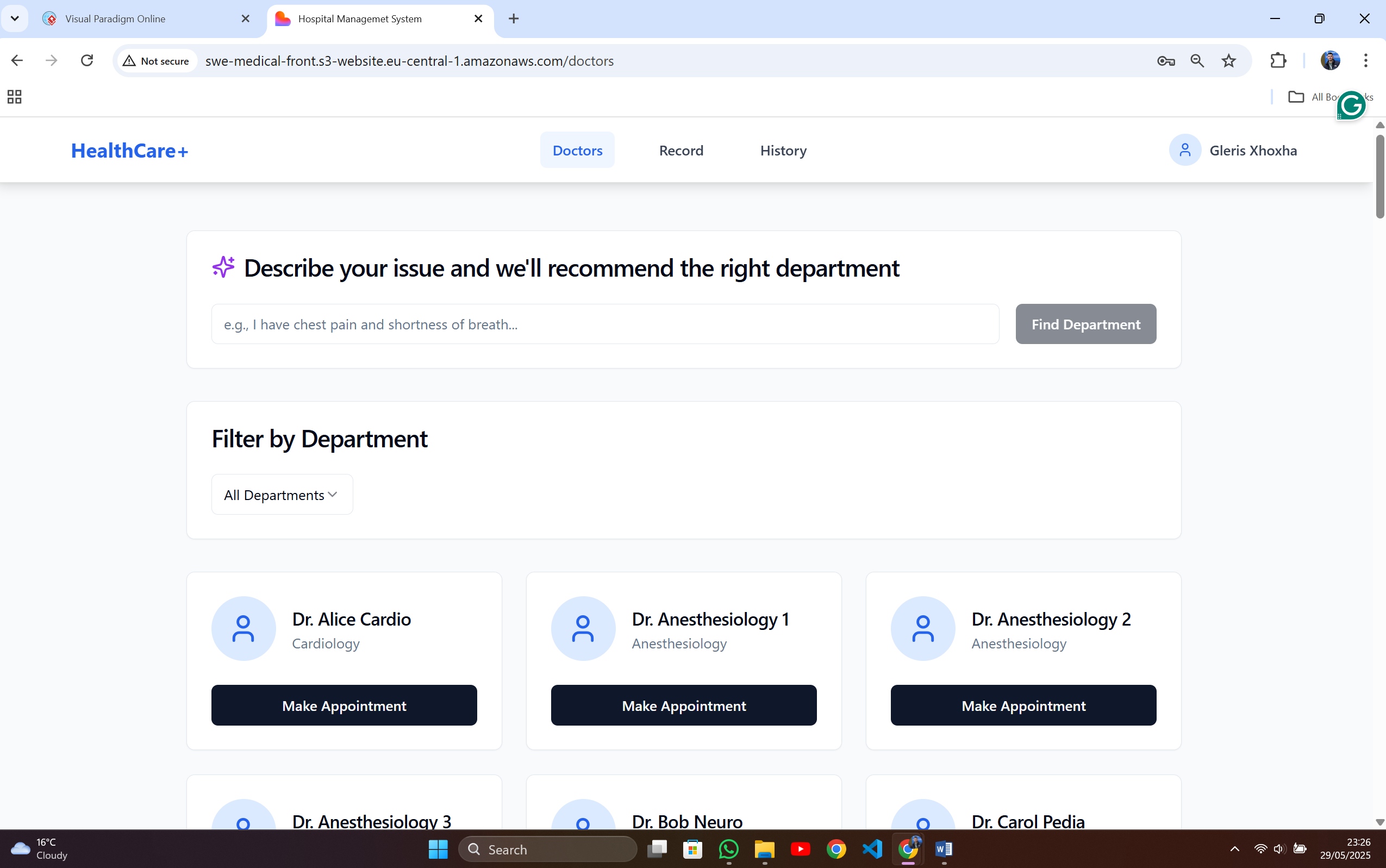
- Tracks project tasks, milestones, and progress.

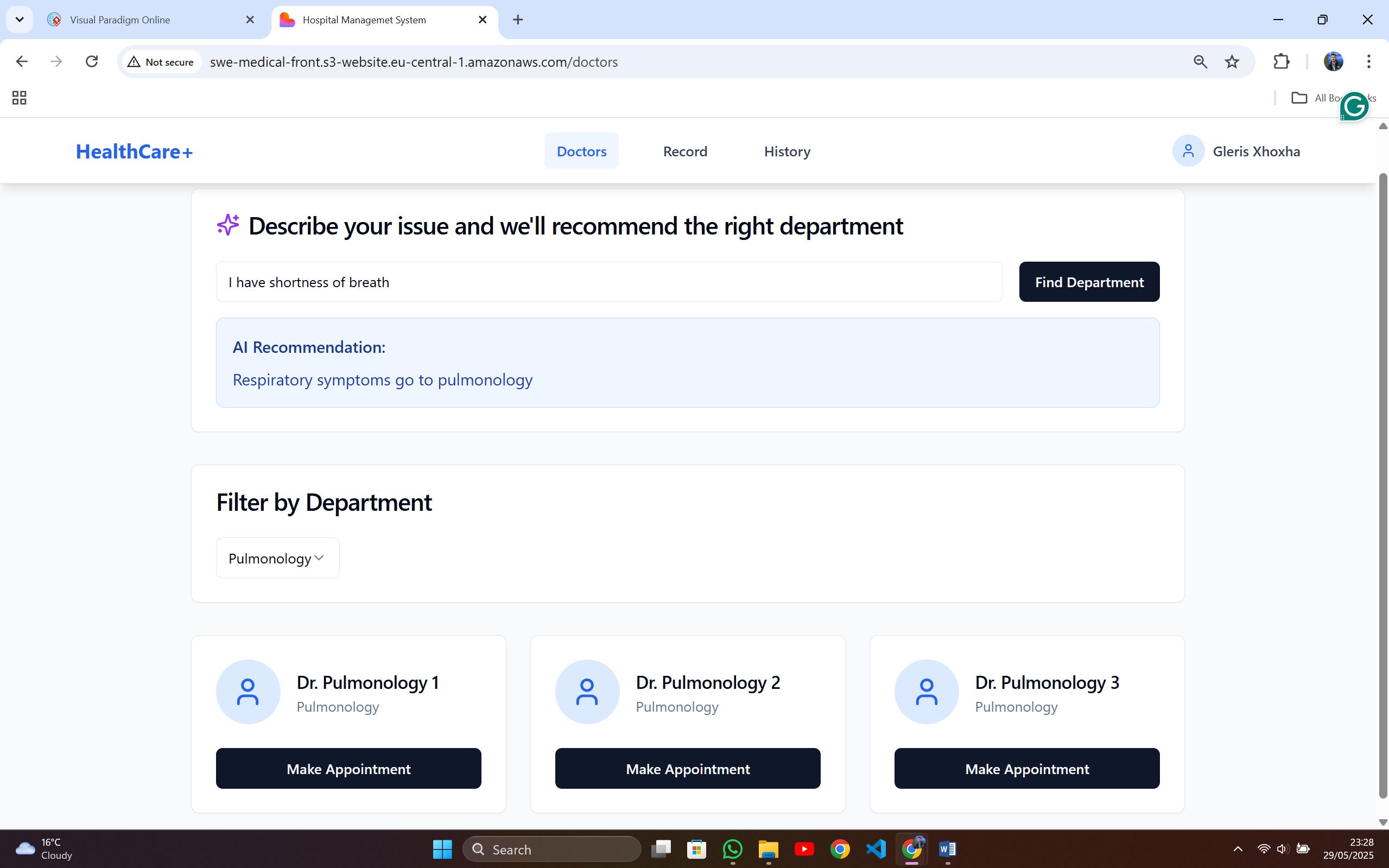
Link for the backend repository : <https://github.com/kallash04/swe-medical-backend>

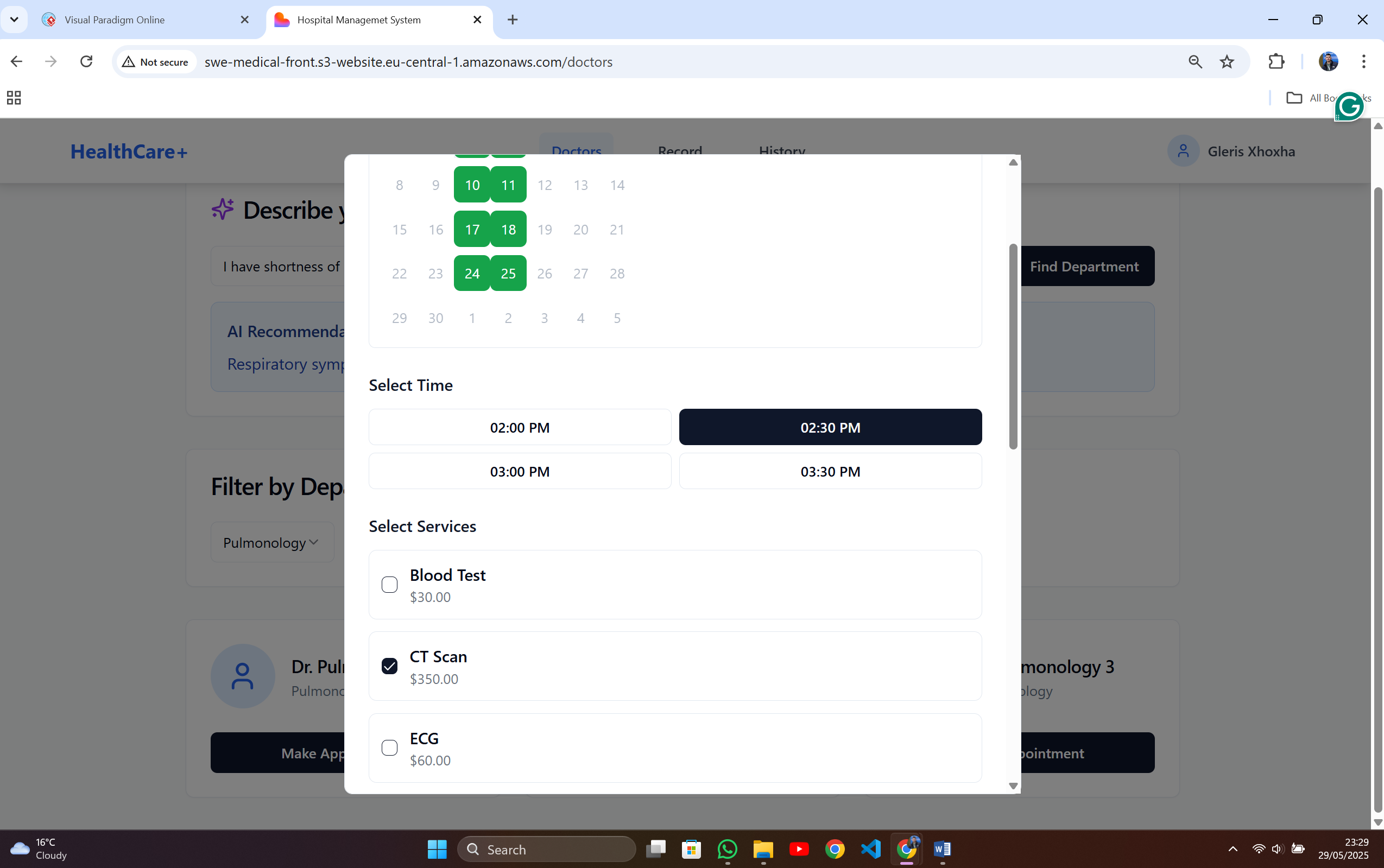
Link for the frontend repository : <https://github.com/kallash04/swe-medical-frontend>

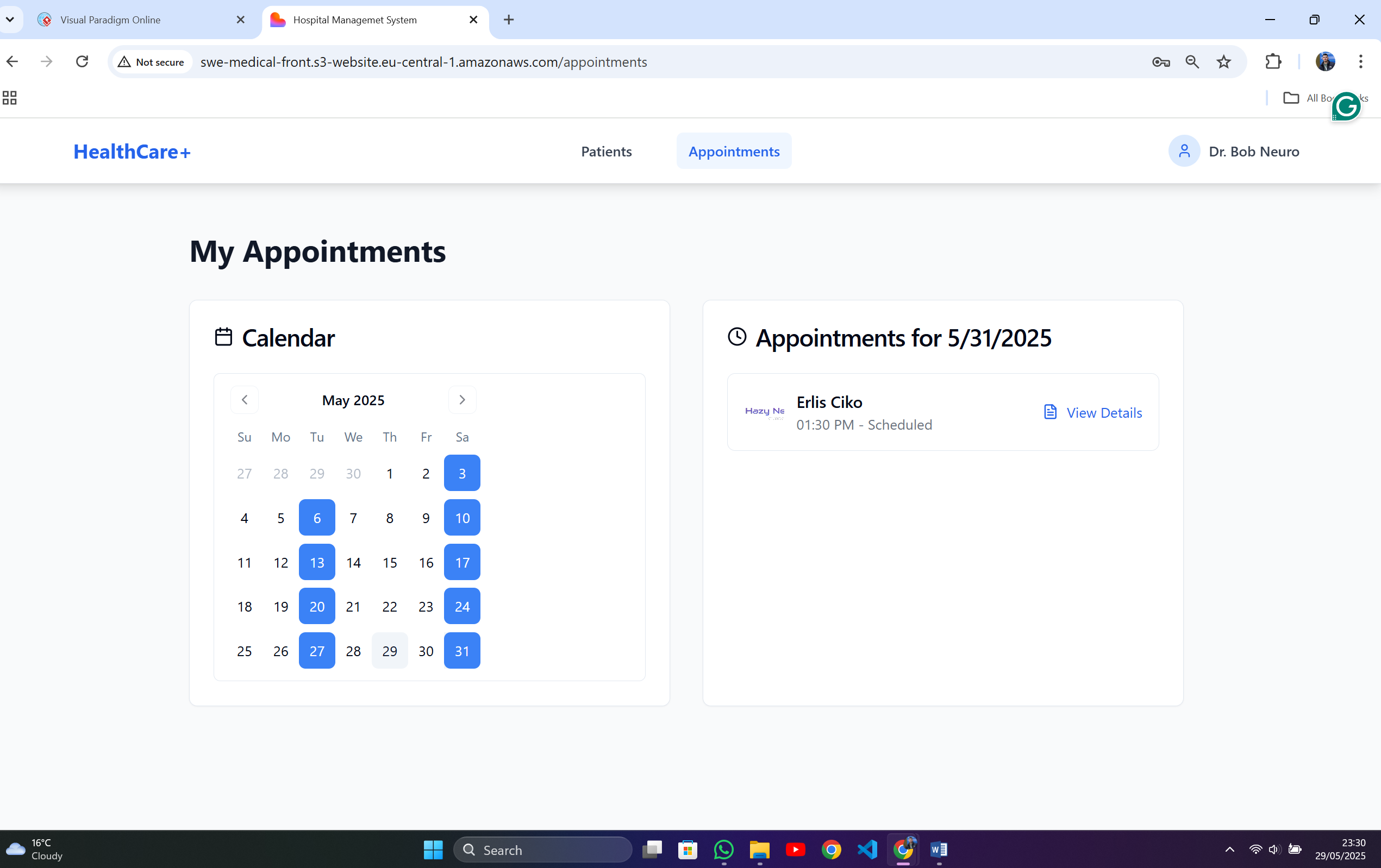
## **5.1 Implementation Images**

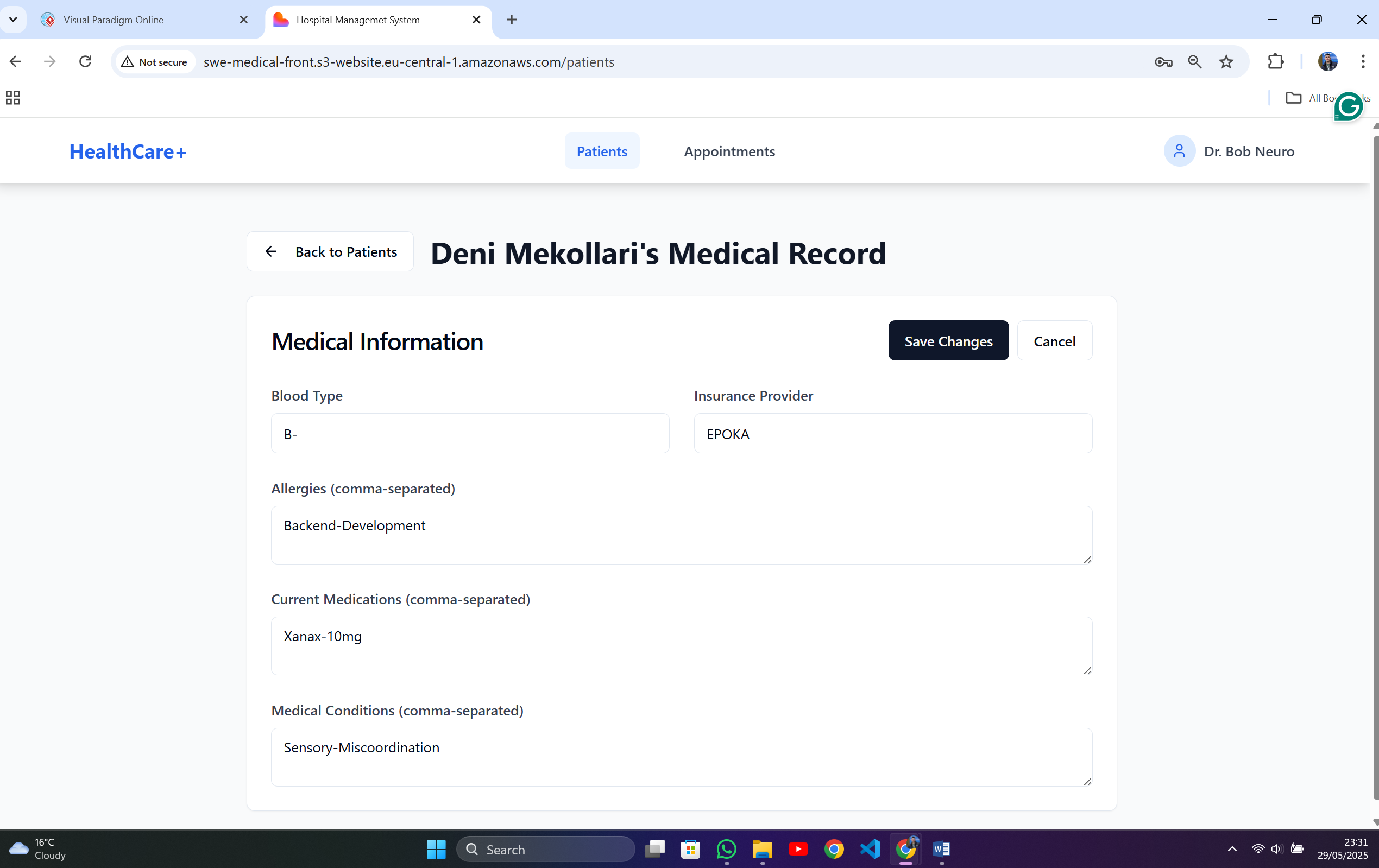
****

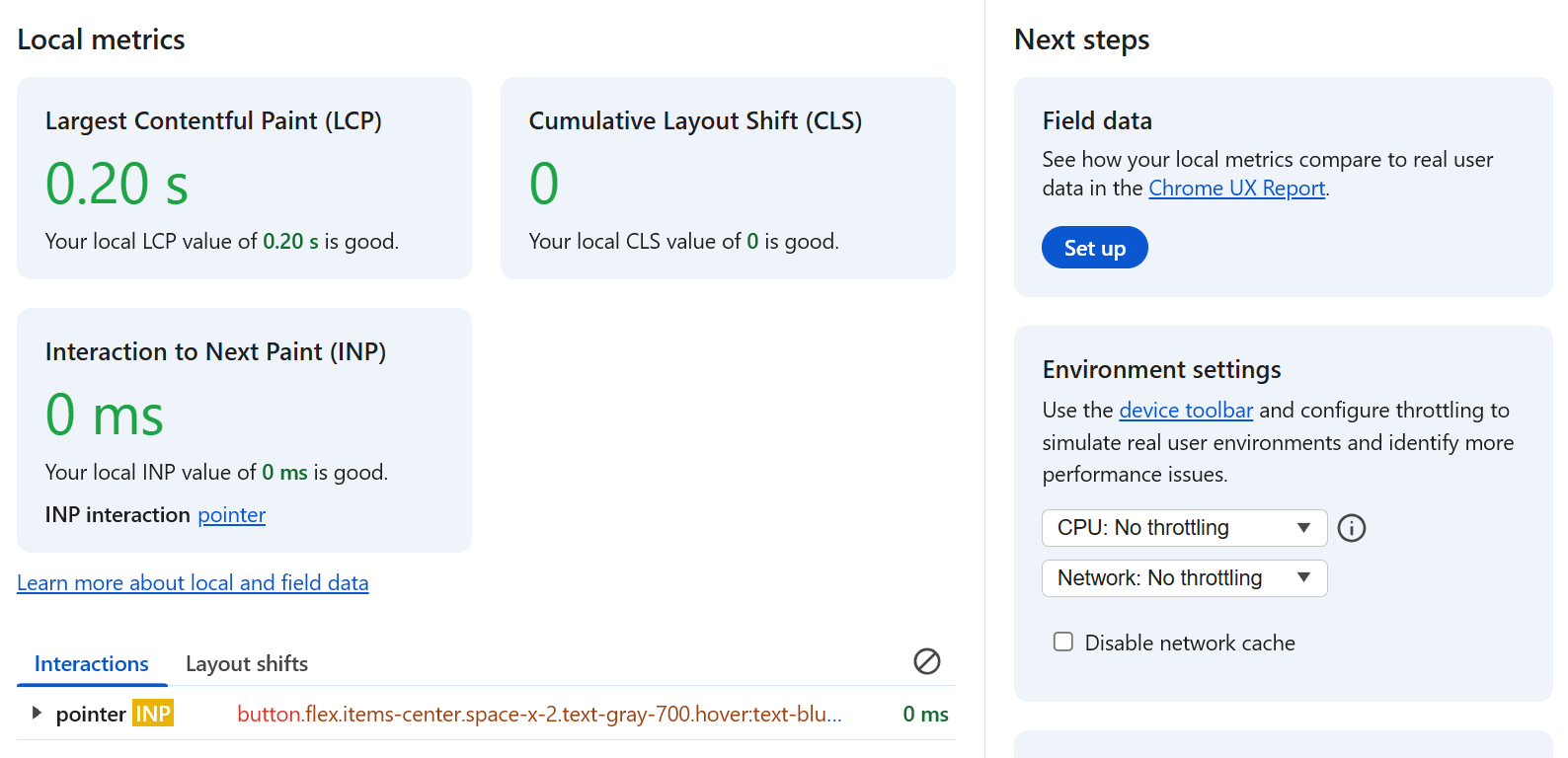
****

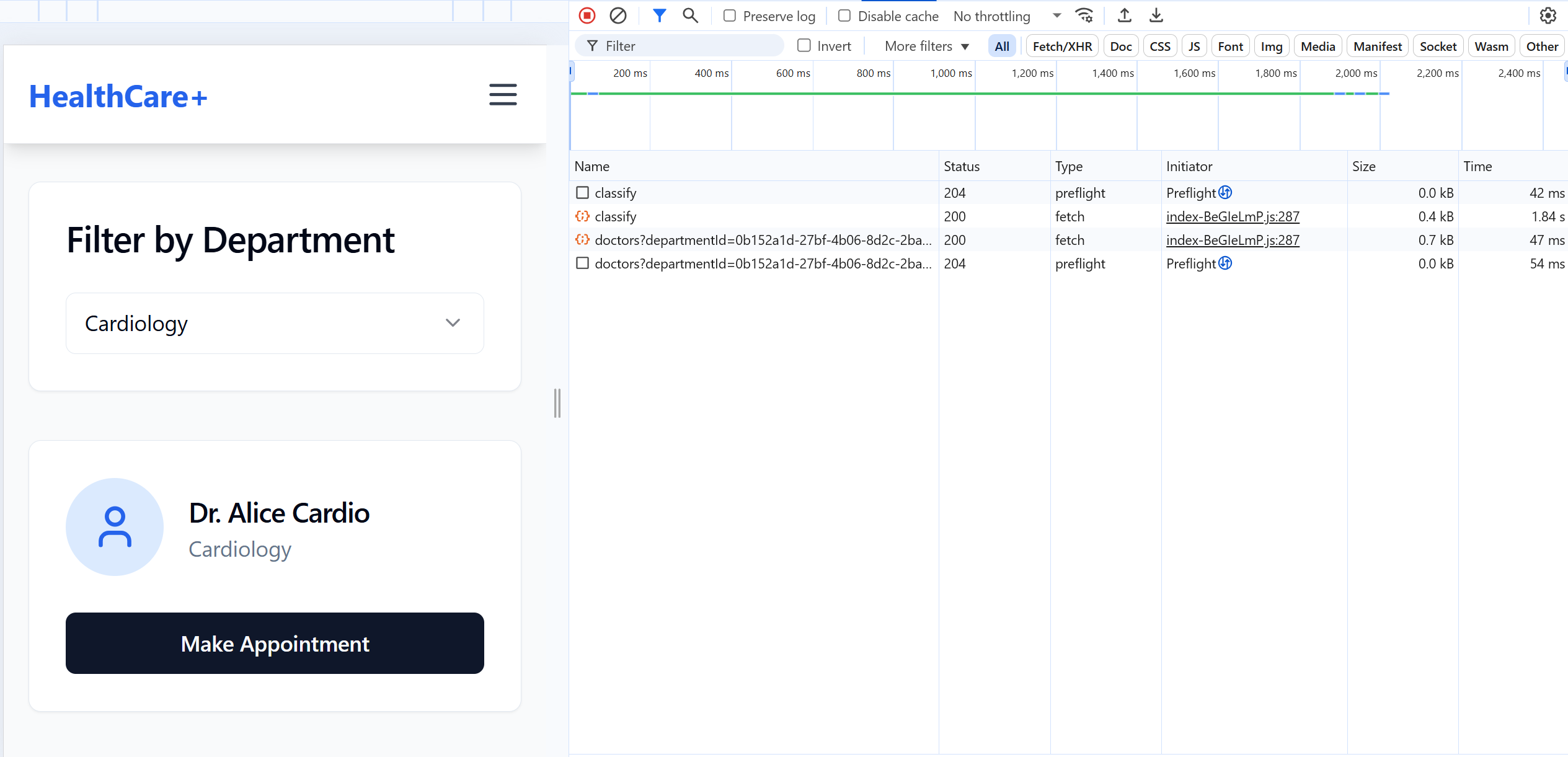
****

****

****

****

****

****