**MEDICAL APPOINTMENT BOOKING SYSTEM**

**A MINI - PROJECT REPORT**

***Submitted by***

|  |  |
| --- | --- |
| **AKASH K** | **Reg. No. 9517202304008** |
| **AKILAN N** | **Reg. No. 9517202304009** |
| **NAVEEN BALAN J** | **Reg. No. 9517202304099** |

***for the Course***

**23CS453 – MINI PROJECT - II**

**Logo

Description automatically generated**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**MEPCO SCHLENK ENGINEERING COLLEGE, SIVAKASI**

**(An Autonomous Institution affiliated to Anna University Chennai)**

**May 2025**

***Register No.* :** .………………………

**BONAFIDE CERTIFICATE**

This is to certify that it is the Bonafide work titled “**Medical Appointment Booking System”** carried out by **Akash K   
(RegNo.: 9517202304008), Akilan N (RegNo.:9517202304009),   
Naveen Balan J (Reg No.:9517202304008)** for **23CS453 - Mini Project–II**, Mepco Schlenk Engineering College (Autonomous), Sivakasi during

**IV semester** in the academic year 2024 – 2025.

**Faculty In-Charge** **Head of the Department**

Submitted for the Practical Examination held at Mepco Schlenk Engineering College (Autonomous), Sivakasi on ……/……/ 20…….

…………………………. …………………………

**Internal Examiner - I** **Internal Examiner - II**

**ABSTRACT**

The Medical Appointment Booking System (MABS) serves as a powerful tool for small businesses, streamlining the scheduling process by offering customers an efficient and user-friendly way to book appointments. By reducing the reliance on phone calls and manual scheduling, this system significantly decreases the administrative workload typically shouldered by staff, allowing them to focus on other important tasks. Moreover, MABS enhances patient satisfaction by providing a seamless and flexible experience for securing appointments.

This modern approach to scheduling empowers businesses to deliver convenience and accessibility to their customers. The implementation of such a system not only boosts operational efficiency but also alleviates the pressure on workers, fostering a more balanced and productive work environment. In essence, MABS bridges the gap between business operations and customer expectations, presenting a win-win scenario that improves service quality, optimizes resource utilization, and promotes a stress-free experience for both staff and customers.

**ACKNOWLEDGEMENT**

First and foremost, we thank the **LORD ALMIGHTY** for his abundant blessings that is showered upon our past, present and future successful endeavors.

We extend our sincere gratitude to our college management and Principal **Dr. S. Arivazhagan M.E., Ph.D.,** for providing sufficient working environment such as systems and library facilities. We also thank him very much for providing us with adequate lab facilities, which enable us to complete our mini-project.

We would like to extend our heartfelt gratitude to **Dr. J. Raja Sekar M.E., Ph.D.,** Professor and Head, Department of Computer Science and Engineering, Mepco Schlenk Engineering College for giving me the golden opportunity to undertake a mini-project of this nature and for his most valuable guidance given at every phase of our work.

We would also like to extend our gratitude and sincere thanks to   
**Mrs. G. Priyanka M.E., (Ph.D.,)** Assistant Professor (Sl. Grade) and **Dr. N. Kavitha, M.E., Ph.D.,** Assistant Professor (Sr. Grade), Department of Computer Science and Engineering, Mepco Schlenk Engineering College for being our Project Mentors. They put their valuable experience and expertise in directing, suggesting and supporting us throughout the mini-project to bring out the best.

Our sincere thanks to our revered **faculty members and lab technicians** for their help over this mini-project work.

Last but not least, we extend our indebtedness towards out beloved family and our friends for their support which made the mini-project a successful one.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT** | **ii** |
|  | **LIST OF TABLES** | **v** |
|  | **LIST OF FIGURES** | **vi** |
| **1** | **INTRODUCTION** | **7** |
| **2** | **EXISTING & PROPOSED MODEL** | **8** |
| **3** | **SYSTEM DESIGN** | **9** |
|  | 3.1 Software Requirements Specification | 9 |
|  | 3.2 UML Diagrams | 14 |
|  | 3.3 Design Components | 19 |
|  | 3.4 Databases Description | 19 |
|  | 3.5 User Interface Design | 24 |
| **4** | **SYSTEM IMPLEMENTATION** | **29** |
| **5** | **RESULTS AND DISCUSSION** | **32** |
|  | 5.1 Test Cases and Results | 32 |
| **6** | **CONCLUSION AND FUTURE ENHANCEMENT(S)** | **37** |

|  |  |  |
| --- | --- | --- |
| **APPENDIX – A** | **SYSTEM REQUIREMENTS** | **38** |
| **APPENDIX – B** | **SOURCE CODE** | **39** |
|  | **REFERENCES** | **84** |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO.** | **TABLE CAPTION** | **PAGE NO.** |
| 3.1 | Appointment Description | 19 |
| 3.2 | Billing Description | 20 |
| 3.3 | Departments Description | 20 |
| 3.4 | Doctors Description | 21 |
| 3.5 | Medical records Description | 21 |
| 3.6 | Patients Description | 22 |
| 3.7 | Staff Description | 23 |
| 3.8 | Users Description | 23 |
| 5.1 | Positive Test Case and result for login | 32 |
| 5.2 | Negative Test Case and result for login | 32 |
| 5.3 | Positive Test Case and result for signup | 33 |
| 5.4 | Negative Test Case and result for signup | 33 |
| 5.5 | Positive Test Case and result for appointment booking | 34 |
| 5.6 | Negative Test Case and result for appointment booking | 34 |
| 5.7 | Positive Test Case and result for patient registration | 35 |
| 5.8 | Negative Test Case and result for patient registration | 35 |
| 5.9 | Positive Test Case and result for doctor registration | 36 |
| 5.10 | Negative Test Case and result for doctor registration | 36 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **FIGURE CAPTION** | **PAGE NO.** |
| 3.1 | Use Case Diagram of Medical Appointment Booking System | 14 |
| 3.2 | Class Diagram of Medical Appointment Booking System | 15 |
| 3.3 | Activity Diagram of Medical Appointment Booking System | 16 |
| 3.4 | Sequence Diagram of Medical Appointment Booking System | 17 |
| 3.5 | Component Diagram of Medical Appointment Booking System | 18 |
| 3.6 | Main Layout | 24 |
| 3.7 | Home Page | 24 |
| 3.8 | Patients Page | 25 |
| 3.9 | Doctors Page | 25 |
| 3.10 | Appointments Page | 26 |
| 3.11 | Departments Page | 26 |
| 3.12 | Staff Page | 27 |
| 3.13 | Medical records Page | 27 |
| 3.14 | Billing Page | 28 |
| 3.15 | Notification Page | 28 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**CHAPTER 1**

**INTRODUCTION**

* 1. **PROBLEM STATEMENT**

Many healthcare providers struggle with the traditional appointment scheduling process, often relying on phone calls and manual entry in calendars. This method can lead to miscommunications, double bookings, and increased administrative burden. Additionally, patients frequently experience frustration due to lack of availability and difficulties in scheduling appointments at their convenience. There is a significant need for a digital solution that allows patients to manage their appointments more effectively while easing the workload on medical staff.

**1.2 OBJECTIVES**

The Primary objective of Medical Appointment Booking System (MABS) is to provide a registration and login for both doctors and users. Doctors can register by giving his necessary details like timings, fee, etc. After successful registration, the doctor can log in by giving username and password. The doctor can view the booking request by users and if he accepts the users requests the status will be shown as booking confirmed to the users. He can also view the feedback given by the users. The users must be registered and log in to book a doctor. The search results will show the list of doctors matching users required criteria and he can select one and send a request. The request will be forwarded to admin and admin forward it to doctor and if he is available, he will send the confirmation, it updates the booking request and says confirmed to the users. The users can view the status in the status tab.

* 1. **SCOPE**

The scope of this system to reduce the need for phone calls and manual scheduling, thereby decreasing administrative workload and enhancing patient satisfaction. By offering a convenient and flexible way for patients to secure appointments, healthcare providers can improve operational efficiency and reduce the pressure on medical staff. This system ensures that patients have easy access to medical services, ultimately contributing to better patient care and resource management.

**CHAPTER 2**

**EXISTING AND PROPOSED MODEL**

**2.1 EXISTING MODEL**

The functional requirement in medical appointment booking System is the collective information about what are the operations available in the system.

* It should provide the basic appointment scheduling through manual phone-calls or in-person visits to healthcare facilities.
* Healthcare staff manually maintains patient records and appointment schedules in physical registers or basic digital systems.
* Patients need to physically visit the healthcare facility to check appointment availability and book slots.
* The system relies on traditional communication methods for appointment reminders and updates.
* Limited ability to handle emergency appointments or lastminute schedule change.
* No real-time visibility of doctor availability or appointment slots.
* Manual processing of appointment cancellations and rescheduling requests.

**2.2 PROPOSED MODEL**

The non-functional requirement describes about platform and physical resource required for building the medical appointment booking system.

* It should provide the functionality of authentication for valid users, including patients and doctors.
* It should allow patients to view available appointment slots for doctors in their preferred location or specialty.
* It should enable patients to book, reschedule, or cancel appointments.
* It should provide doctors with functionality to view and manage their schedules and appointments.
* It should allow notifications to be sent to patients and doctors regarding appointment confirmations, reschedules, or cancellations.

**CHAPTER 3**

**SYSTEM DESIGN**

**3.1 SOFTWARE REQUIREMENTS SPECIFICATION**

**3.1.1 Introduction**

**System Overview:**

There are three Main Module in this appointment booking system. They are:

**a) Admin:**

Admin needs to login with username and password and in the admin home screen, he can see the basic functionalities of admin. Admin can add doctors and view doctor. He can also view the user’s request. Admin can view the appointment and view feedback of users. Admin can able to control the whole system. He/she can add, delete, update and modify the system. Admin keeps the system up-to-date.

**b) Doctor:**

Doctor needs to be registered by giving the necessary details like experience, timing, fees etc. After registering, he needs to log in and in the home screen he can view the basic functionalities. Doctors are able to see the respective appointments taken. Doctor should fairly know about the usage of the system. He can view and accept the user request.

**c) Users:**

The user needs to be registered and log in after logging on he can search for the doctor by giving the location. Based on the doctor availability, the booking request will be confirmed. He can also view in the status and he can also give feedback biasing the performance of the doctor.

**3.1.2 Tools/Environment used**

Memory : 2GB RAM or above.

Cache Memory : 512 KB or above.

Software Operating System : Windows 10, Window 11.

Front end tool : HTML, JavaScript, CSS.

Back end : MySQL

**3.1.3 Functional requirements**

**a) Home Page**

It shows the basic view of the application home page and the list of modules.

**b) Admin Login Page**

In this page, admin can log in by giving username and password. After successful login, the application shows the admin home page in which the basic functionalities are shown.

**c) Add/Delete/Update/Modify records:**

Admin can add doctor, view doctor, view users, view appointment, view feedback.

I. Add Doctor:

Admin can add doctor details like doc id, name, address, mobile number, email, category.

II. View Doctor:

Admin can view doctor list.

III. View users:

Admin can view user’s details like name, email, phone, gender, age, address.

IV. View Appointment:

Admin can view appointment like user id, doctor name, appointment date, appointment time.

V. View Feedback:

Admin can view feedback details like user id, Feedback (Good, Bad), date

VI. Add Walk-In Users:

Admin can add walk-in users and adjust users according to online appointment users. Add Walk-In User’s Name Email-Id Phone Gender Age Address Time

**d)Users login:**

In user details, first users registered for online appointment, after registration process user login with username and password. User can book online appointment, view booking, cancel booking, search doctor, give feedback.

I. User registration process:

In this page, the users can register into the application by providing all necessary details.

II. User login:

User login with username and password for online appointment.

III. Book Appointment:

User book appointment with appointment Id, category, select doctor, select date, select time and other details.

IV. Cancel booking:

User can cancel online booking appointment.

V. View booking:

Users can view booking history.

VI. Feedback:

Users can give the feedback.

**e) Doctor login:**

Doctor login with user-id and password, he/she can view appointment details, view user details.

I. Doctor login:

Doctor login with username and password.

II. My appointment:

Doctor view appointments details.

III. View user details:

Doctor can view user details.

**3.1.4 Non-Functional Requirements for Appointment Booking System:**

**a) Performance Requirements:**

1. Response Time: The system should respond to user requests within 2 seconds for 95% of transactions.

2. Scalability: The system should handle up to 10,000 concurrent users without performance degradation.

3. Capacity: The system should support the storage of up to 1 million appointment records.

**b) Reliability Requirements:**

1. Availability: The system should have an uptime of 99.9% to ensure it is available to users at all times.

2. Backup and Recovery: The system should perform daily backups and be able to recover from failures within 4 hours.

**c) Security Requirements:**

1. Authentication: The system should require secure login for all user types (admin, doctor, and user).

2. Authorization: The system should ensure that users can only access data and perform actions that they are authorized for.

3. Data Encryption: All sensitive data should be encrypted in transit and at rest.

4. Audit Logging: The system should maintain an audit log of all user activities for security and compliance purposes.

**d) Usability Requirements:**

1. User Interface: The system should have an intuitive and user-friendly interface for all user types.

2. Accessibility: The system should comply with certain standards to ensure accessibility for users with disabilities.

3. Documentation: Comprehensive user manuals and help guides should be available for all user types.

**e) Maintainability Requirements:**

1. Code Quality: The system should follow coding standards and best practices to ensure code quality and readability.

2. Modularity: The system should be designed in a modular way to facilitate easy updates and maintenance.

3. Documentation: The system should have detailed technical documentation for developers.

**f) Portability Requirements:**

1. Platform Independence: The system should be platform-independent and run on any operating system that supports the required environment.

2. Browser Compatibility: The system should be compatible with major web browsers including Chrome, Firefox, Safari, and Edge.

**g) Compliance Requirements:**

1. Data Protection: The system should comply with relevant data protection regulations such as GDPR.

2. Healthcare Regulations: The system should comply with healthcare regulations such as HIPAA for handling patient data.

**h) Performance Monitoring Requirements:**

1. Monitoring Tools: The system should include tools for monitoring performance, including response times, error rates, and system load.

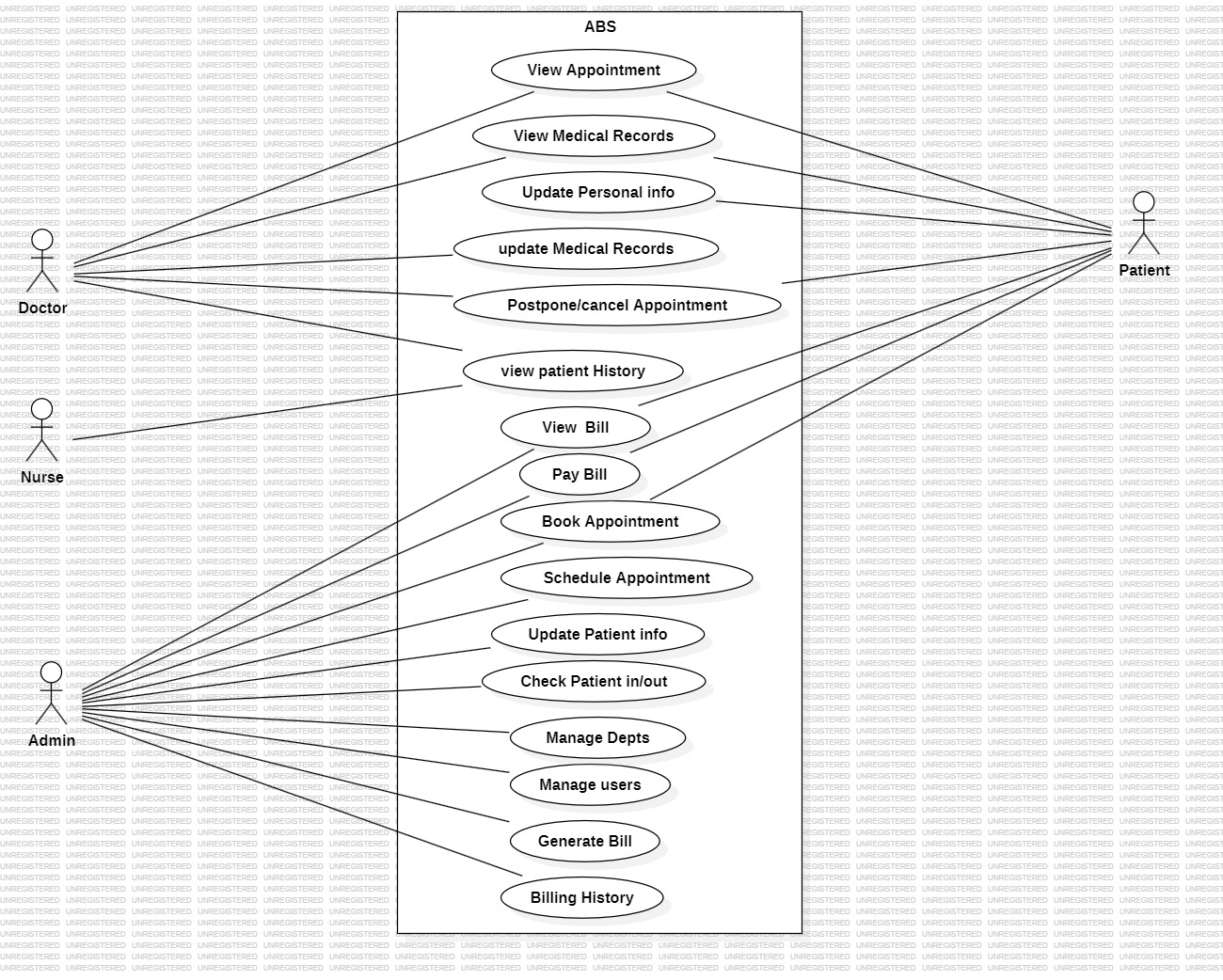
2. Alerts: The system should have alert mechanisms in place to notify administrators of potential issues.

**i) Environmental Requirements:**

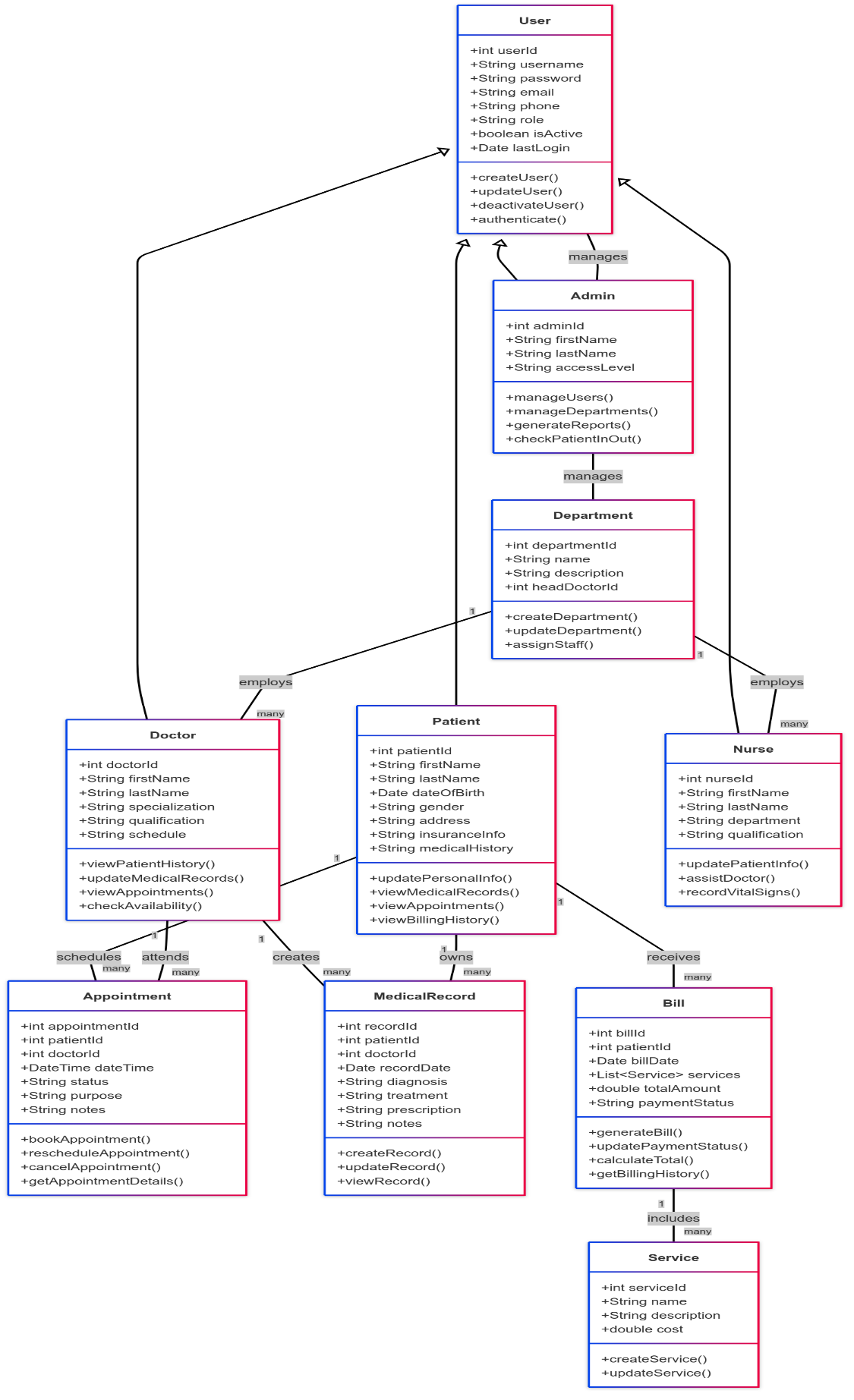
1. Operating Environment: The system should operate efficiently in a web environment with varying network conditions.

2. Hardware Requirements: The system should operate on hardware with a minimum of 256MB RAM and 128 KB cache memory.

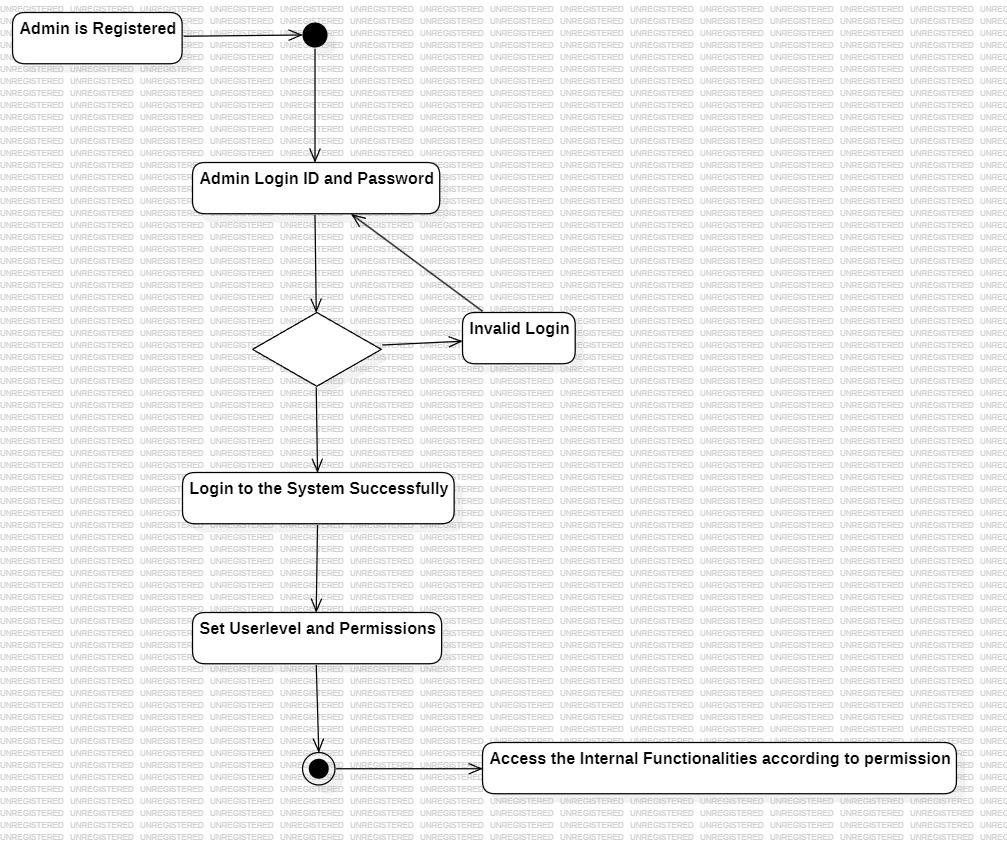
**3.2 UML Diagrams**

****

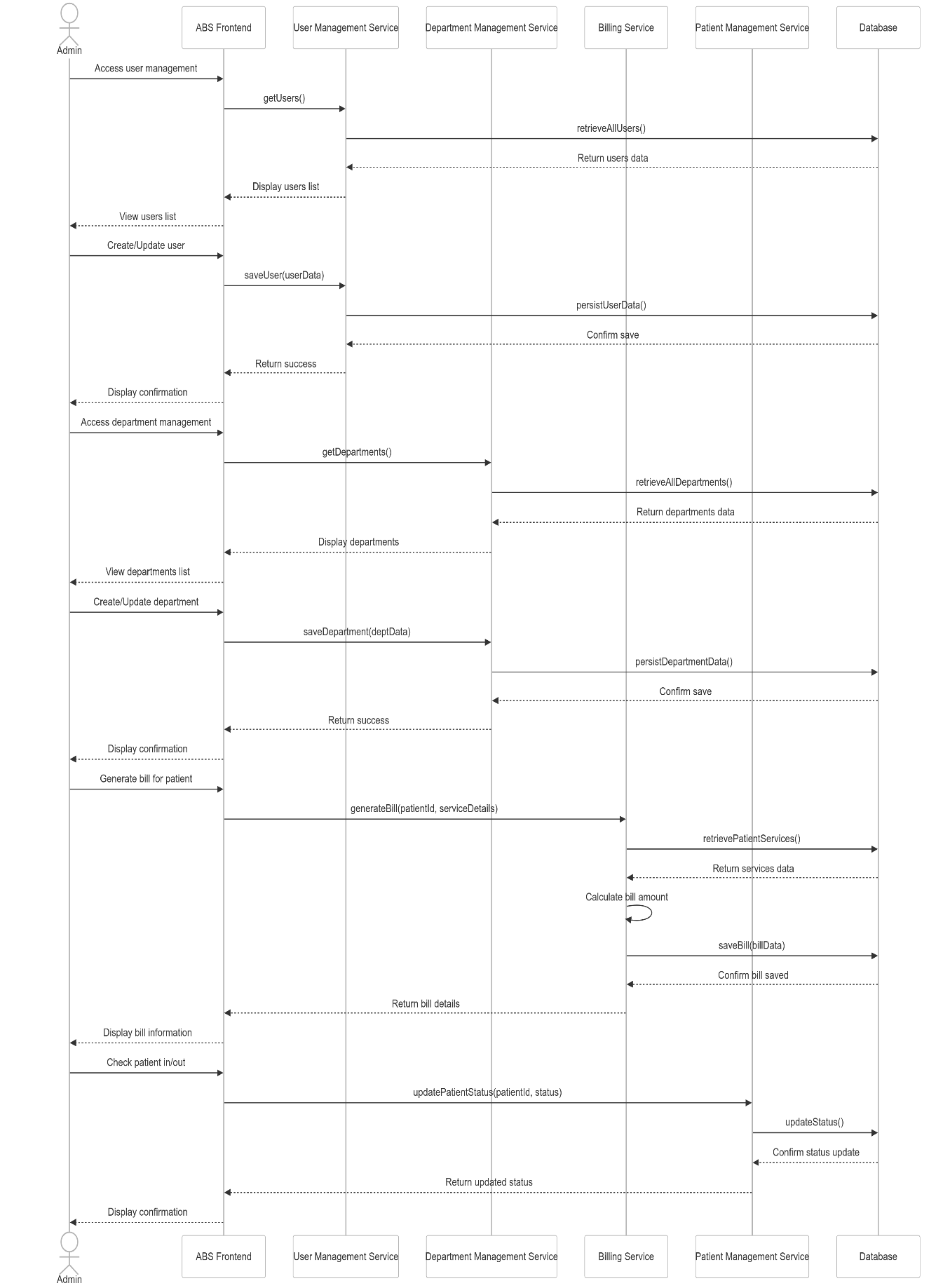
**Figure 3.1: Use Case Diagram for Medical booking appointment system**

****

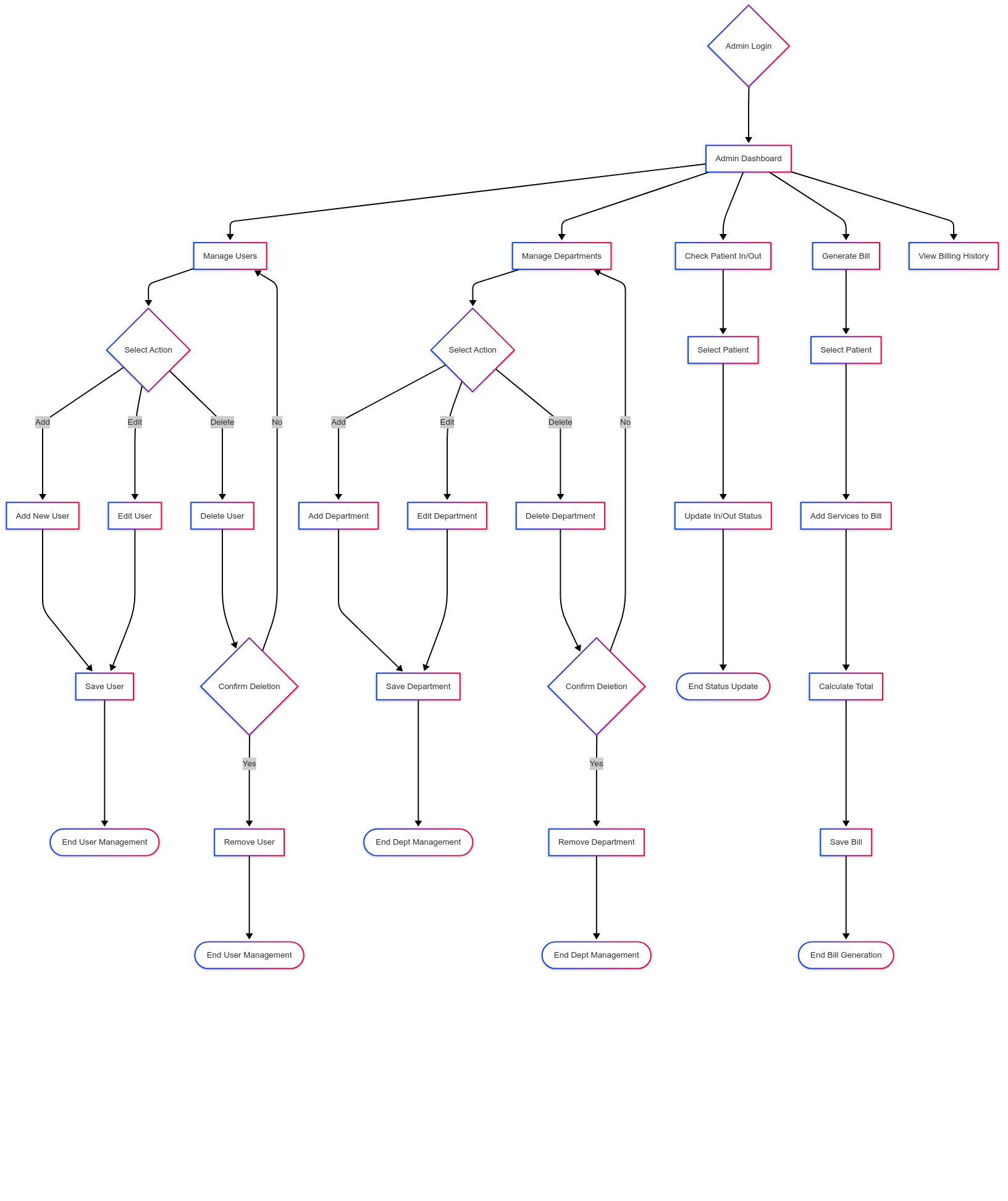
**Figure 3.2: Class Diagram for Medical appointment booking system**

****

**Figure 3.3: Activity Diagram for Medical booking appointment system**

****

**Figure 3.4: Sequence diagram for medical booking appointment system**

**Figure 3.5: Component diagram for medical appointment booking system**

**3.3 DESIGN COMPONENTS**

**3.3.1 Front End:**

The Medical Appointment Booking System (MABS) uses HTML, JavaScript, CSS for developing interactive pages.

**3.3.2 Back End:**

Uses MySQL for back end to store data.

**3.4 DATABASE DESCRIPTION**

**3.4.1 Appointments Structure**

As shown in table 3.1, appointments structure contains the details of patient appointments.

| **Attribute Name** | **Type** | **Width** | **Constraint(s)** | **Description** |
| --- | --- | --- | --- | --- |
| Id | Integer |  | Primary Key, Auto Increment | Unique identifier for the appointment |
| Name | String | 255 | Not Null | Name of the patient making the appointment |
| Email | String | 255 | Not Null | Email address of the patient |
| Phone | String | 20 | Not Null | Contact phone number of the patient |
| department | String | 255 | Not Null | Department where the appointment is scheduled |
| doctor\_id | Integer |  | Foreign Key, Not Null | Reference to the doctor handling the appointment |
| Date | Date |  | Not Null | Date of the appointment |
| Time | Time |  | Not Null | Time of the appointment |
| Reason | Text |  |  | Reason for the appointment |

**Table 3.1: Appointments Description**

**3.4.2 Billing Structure**

As shown in table 3.2, billing structure contains the payment details related to appointments.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Name** | **Type** | **Width** | **Constraint(s)** | **Description** |
| Id | Integer |  | Primary Key, Auto Increment | Unique identifier for the billing record |
| appointment\_id | Integer |  | Not Null | Reference to the associated appointment |
| patient\_name | String | 255 | Not Null | Name of the patient being billed |
| doctor\_name | String | 255 | Not Null | Name of the doctor providing service |
| appointment\_date | Date |  | Not Null | Date of the appointment being billed |
| billing\_date | Date |  | Not Null | Date when the bill was generated |
| service\_details | Text |  |  | Details of services provided |
| Amount | Decimal | 10,2 | Not Null | Initial amount before discount and tax |
| Discount | Decimal | 10,2 | Default 0.00 | Discount applied to the bill |
| Tax | Decimal | 10,2 | Default 0.00 | Tax applied to the bill |
| total\_amount | Decimal | 10,2 | Not Null | Final amount after discount and tax |
| Status | Enum |  | Not Null | Payment status (Paid, Unpaid, Pending) |
| payment\_method | String | 50 |  | Method used for payment |

**Table 3.2: Billing Description**

**3.4.3 Departments Structure**

As shown in table 3.4, departments structure contains details of hospital departments.

| **Attribute Name** | **Type** | **Width** | **Constraint(s)** | **Description** |
| --- | --- | --- | --- | --- |
| Id | Integer |  | Primary Key, Auto Increment | Unique identifier for the department |
| Name | String | 255 | Not Null | Name of the department |
| Head | String | 255 |  | Name of the department head |
| Location | String | 255 |  | Physical location of the department |
| extension | String | 20 |  | Phone extension for the department |
| description | Text |  |  | Detailed description of the department |

**Table 3.4: Departments Description**

**3.4.4 Doctors Structure**

As shown in table 3.5, doctors structure contains details of the medical practitioners.

| **Attribute Name** | **Type** | **Width** | **Constraint(s)** | **Description** |
| --- | --- | --- | --- | --- |
| Id | Integer |  | Primary Key, Auto Increment | Unique identifier for the doctor |
| first\_name | String | 255 | Not Null | First name of the doctor |
| last\_name | String | 255 | Not Null | Last name of the doctor |
| email | String | 255 | Unique, Not Null | Email address of the doctor |
| phone | String | 20 | Not Null | Contact phone number of the doctor |
| department | String | 255 | Not Null | Department where the doctor works |
| specialization | String | 255 | Not Null | Doctor's area of specialization |
| license\_number | String | 50 | Unique, Not Null | Doctor's medical license number |
| join\_date | Date |  | Not Null | Date when the doctor joined the facility |
| consultation\_fee | Decimal | 10,2 | Not Null | Doctor's consultation fee |
| status | Enum |  | Not Null, Default 'active' | Current status (active, on-leave, inactive) |
| qualifications | Text |  |  | Doctor's educational qualifications |
| Bio | Text |  |  | Brief biography of the doctor |

**Table 3.5: Doctors Description**

**3.4.5 Medical Records Structure**

As shown in table 3.6, medical records structure contains patient diagnosis and treatment information.

| **Attribute Name** | **Type** | **Width** | **Constraint(s)** | **Description** |
| --- | --- | --- | --- | --- |
| Id | Integer |  | Primary Key, Auto Increment | Unique identifier for the medical record |
| patient\_id | Integer |  | Foreign Key, Not Null | Reference to the patient |
| doctor\_id | Integer |  | Foreign Key, Not Null | Reference to the doctor who treated the patient |
| diagnosis | Text |  | Not Null | Medical diagnosis of the patient |
| treatment | Text |  |  | Treatment prescribed to the patient |
| Date | Date |  | Not Null | Date when the medical record was created |

**Table 3.6: Medical Records Description**

**3.4.6 Patients Structure**

As shown in table 3.7, patients structure contains detailed information about registered patients.

| **Attribute Name** | **Type** | **Width** | **Constraint(s)** | **Description** |
| --- | --- | --- | --- | --- |
| Id | Integer |  | Primary Key, Auto Increment | System-generated unique identifier |
| patient\_id | String | 20 | Unique, Not Null | Hospital-assigned patient identifier |
| first\_name | String | 50 | Not Null, MUL | First name of the patient |
| last\_name | String | 50 | Not Null | Last name of the patient |
| email | String | 100 | Not Null | Email address of the patient |
| phone | String | 20 | Not Null | Contact phone number of the patient |
| date\_of\_birth | Date |  | Not Null | Patient's date of birth |
| gender | Enum |  | Not Null | Gender (Male, Female, Other) |
| blood\_group | String | 5 |  | Patient's blood group |
| insurance\_number | String | 50 |  | Patient's health insurance number |
| address | Text |  |  | Residential address of the patient |
| medical\_history | Text |  |  | Patient's previous medical history |
| allergies | Text |  |  | Known allergies of the patient |
| created\_at | Timestamp |  | Default CURRENT\_TIMESTAMP | Timestamp when the record was created |
| updated\_at | Datetime |  |  | Timestamp when the record was last updated |

**Table 3.7: Patients Description**

**3.4.7 Staff Structure**

As shown in table 3.8, staff structure contains details of non-doctor employees.

| **Attribute Name** | **Type** | **Width** | **Constraint(s)** | **Description** |
| --- | --- | --- | --- | --- |
| Id | Integer |  | Primary Key, Auto Increment | Unique identifier for the staff member |
| first\_name | String | 255 | Not Null | First name of the staff member |
| last\_name | String | 255 | Not Null | Last name of the staff member |
| email | String | 255 | Unique, Not Null | Email address of the staff member |
| phone | String | 20 | Not Null | Contact phone number of the staff member |
| department | String | 255 | Not Null | Department where the staff member works |
| Role | String | 255 | Not Null | Role of the staff member in the facility |
| specialization | String | 255 |  | Staff member's area of specialization |
| join\_date | Date |  | Not Null | Date when the staff member joined the facility |
| qualifications | Text |  |  | Staff member's educational qualifications |

**Table 3.8: Staff Description**

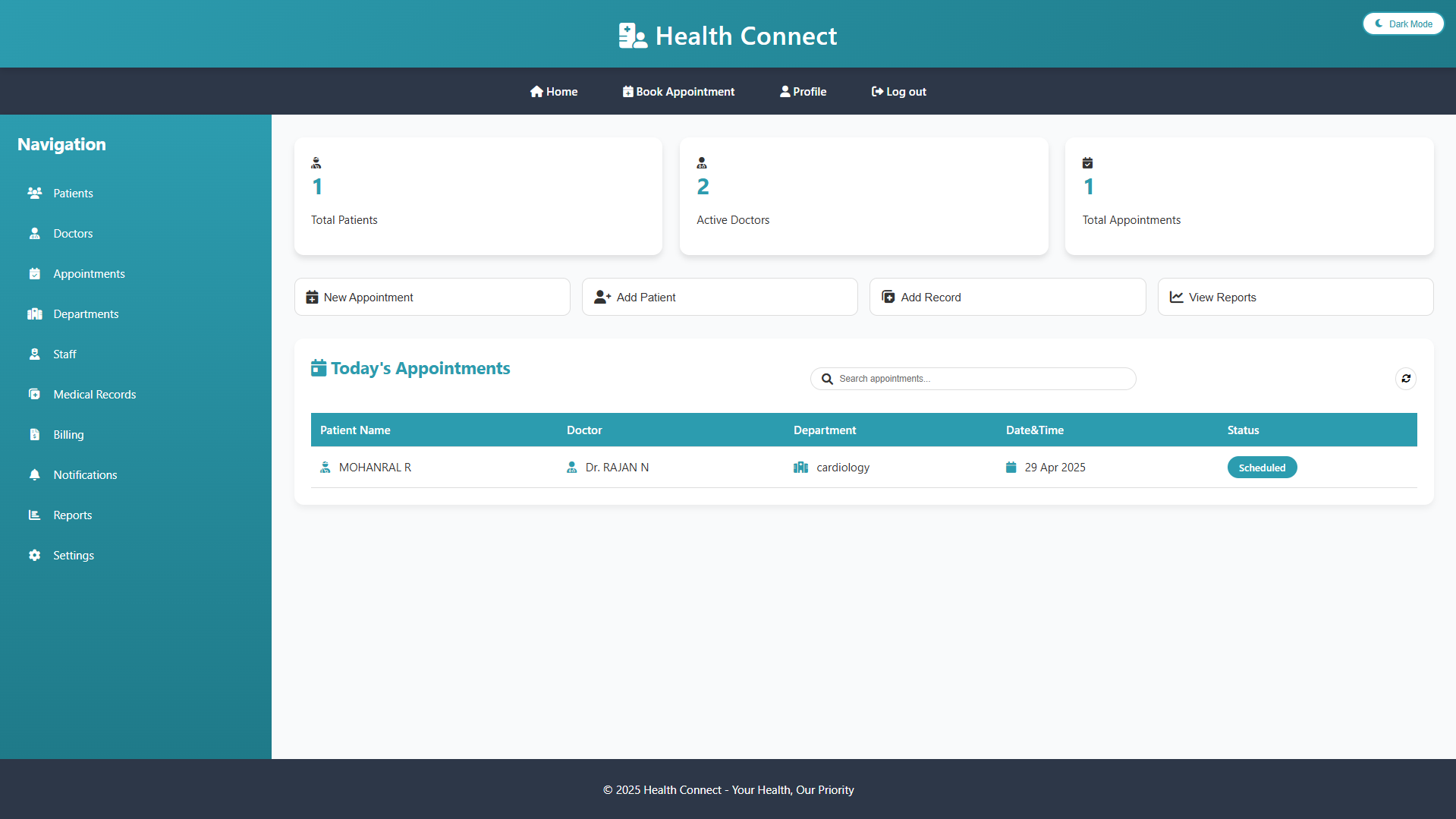
**3.4.8 Users Structure**

As shown in table 3.9, users structure contains login information for system access.

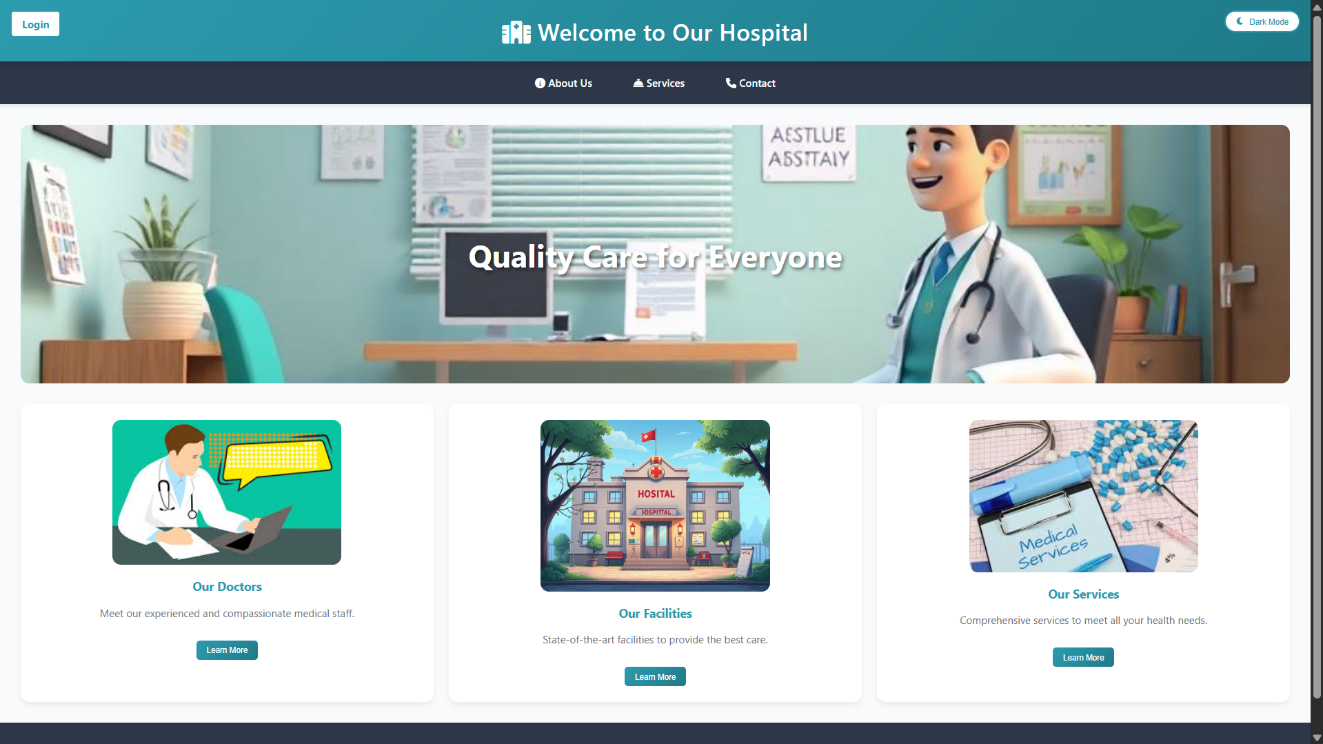
| **Attribute Name** | **Type** | **Width** | **Constraint(s)** | **Description** |
| --- | --- | --- | --- | --- |
| Id | Integer |  | Primary Key, Auto Increment | Unique identifier for the user |
| username | String | 255 | Not Null | Username for system login |
| email | String | 255 | Not Null | Email address associated with the account |
| password | String | 255 | Not Null | Encrypted password for authentication |
| name | String | 255 | Not Null | Full name of the user |

**Table 3.9: Users Description**

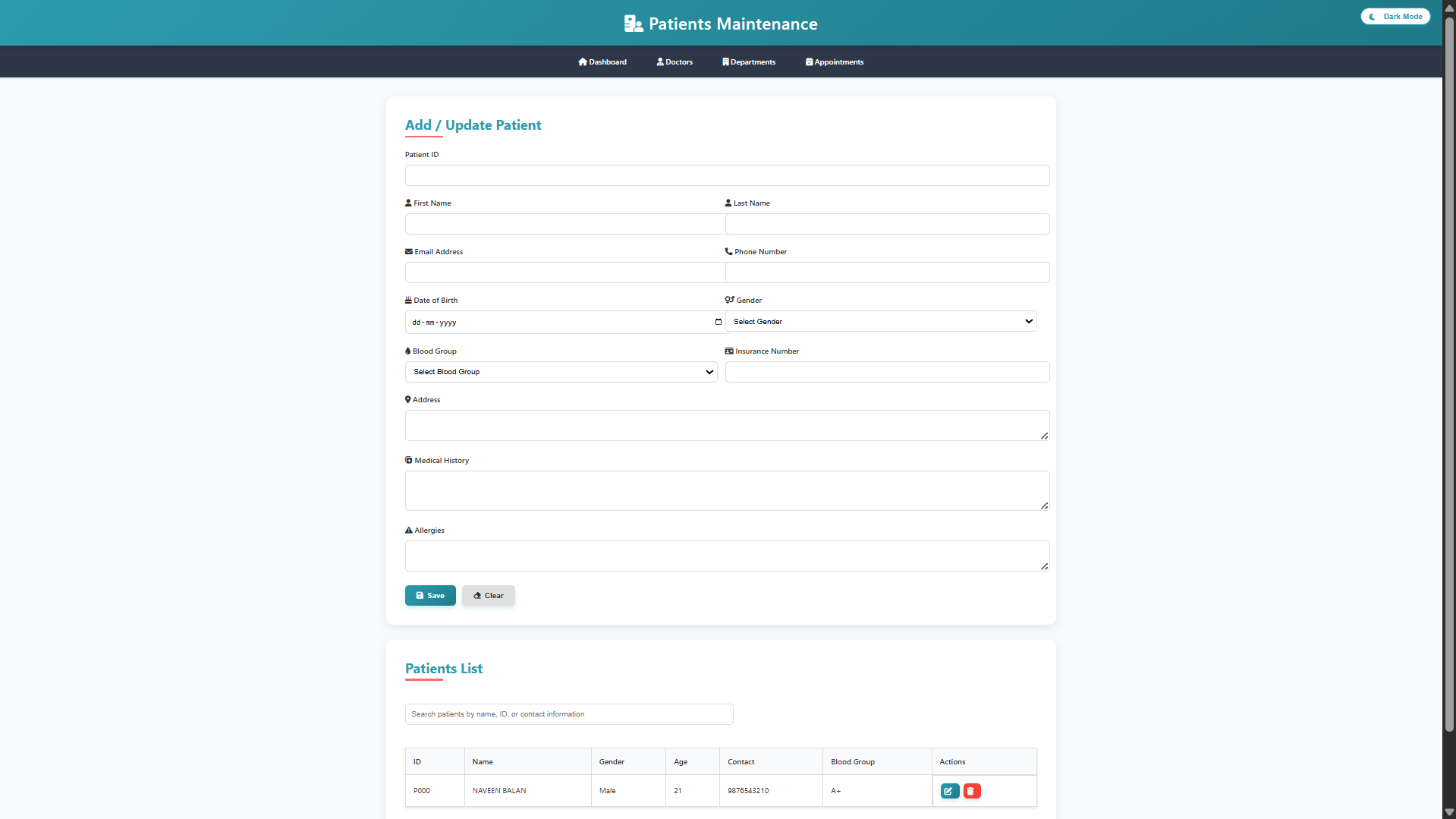
**3.5 USER INTERFACE DESIGN**



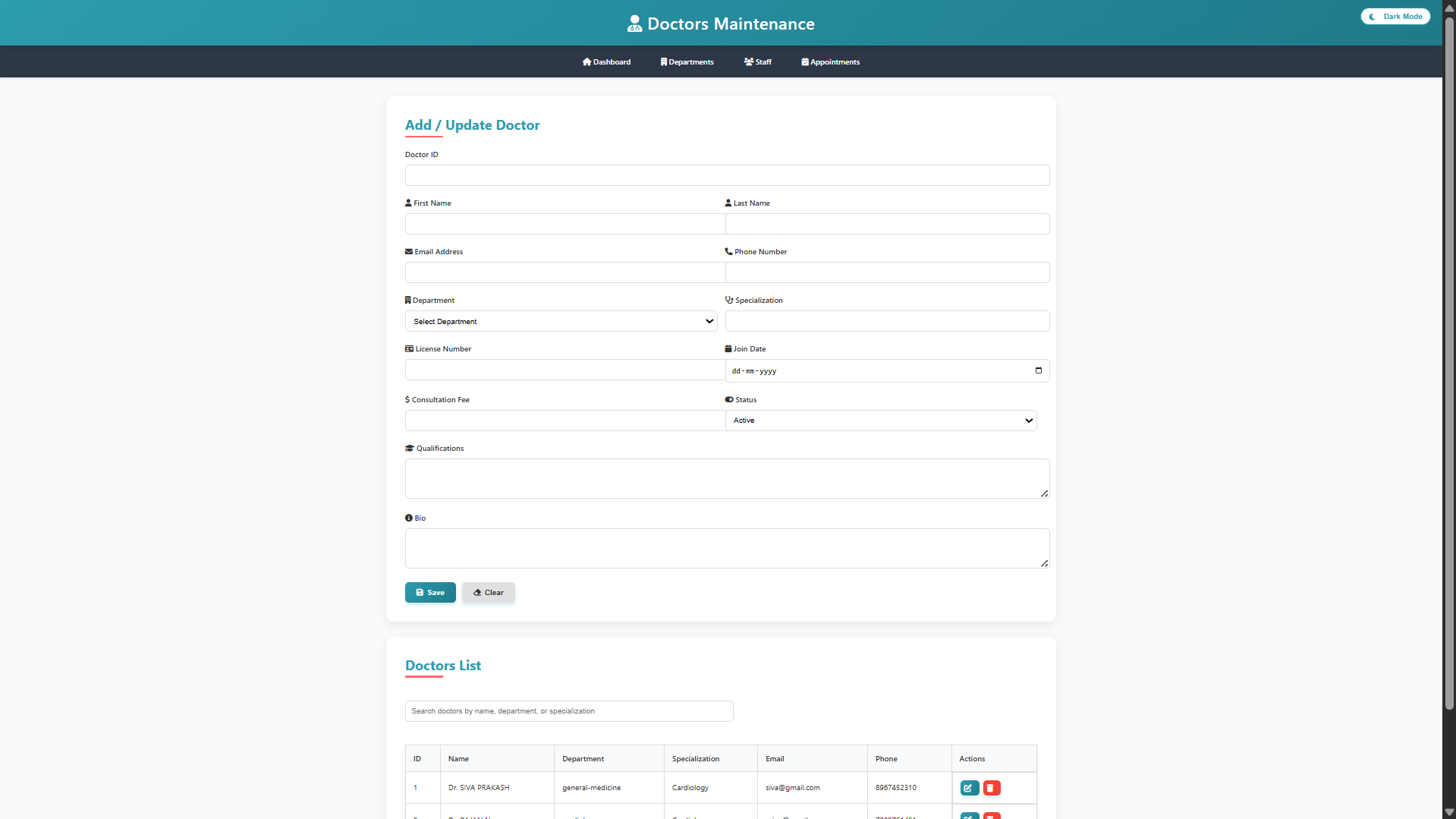
**Figure 3.6: Main layout of Medical Appointment Booking System**



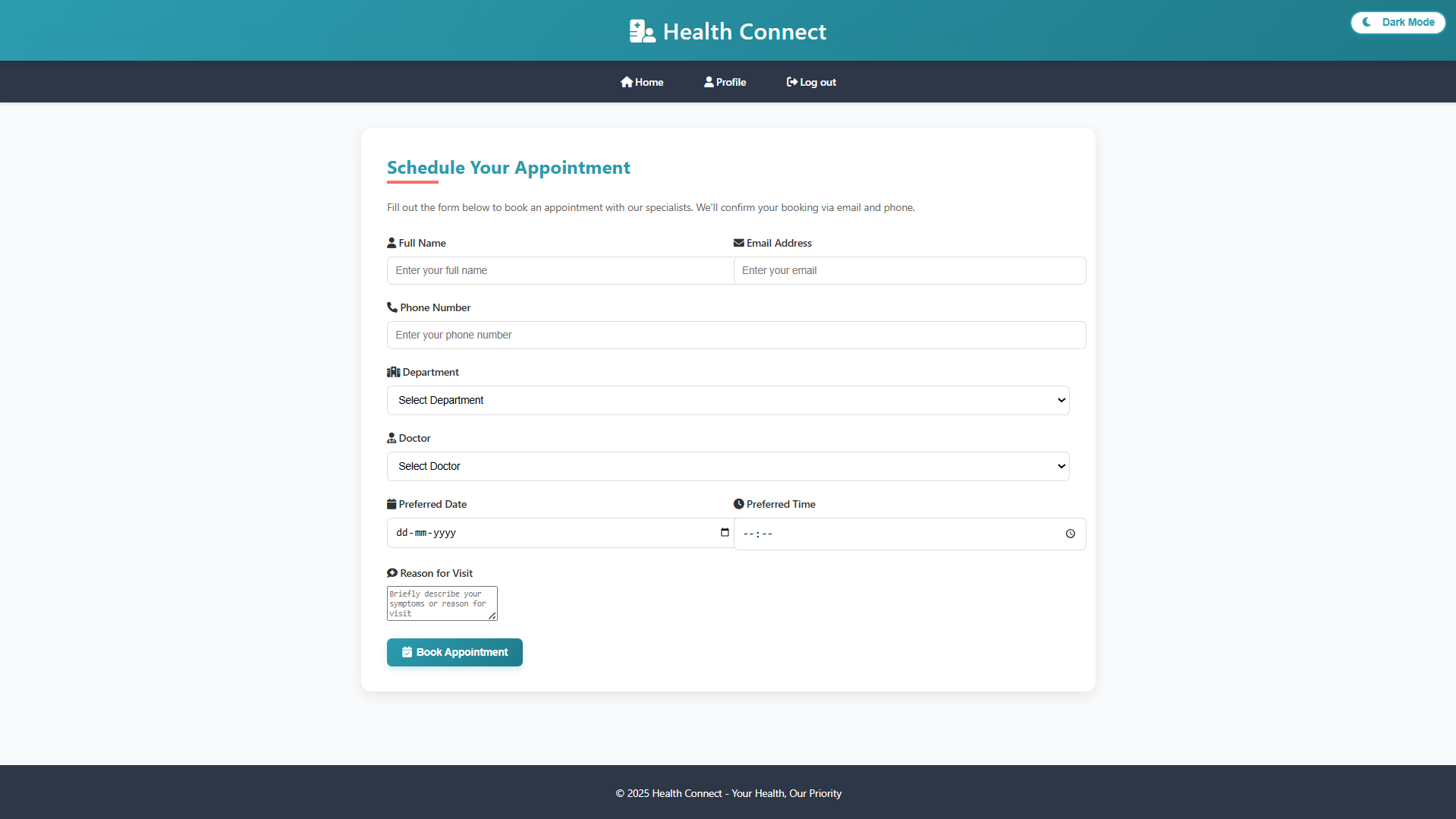
**Figure 3.7: Home Page**



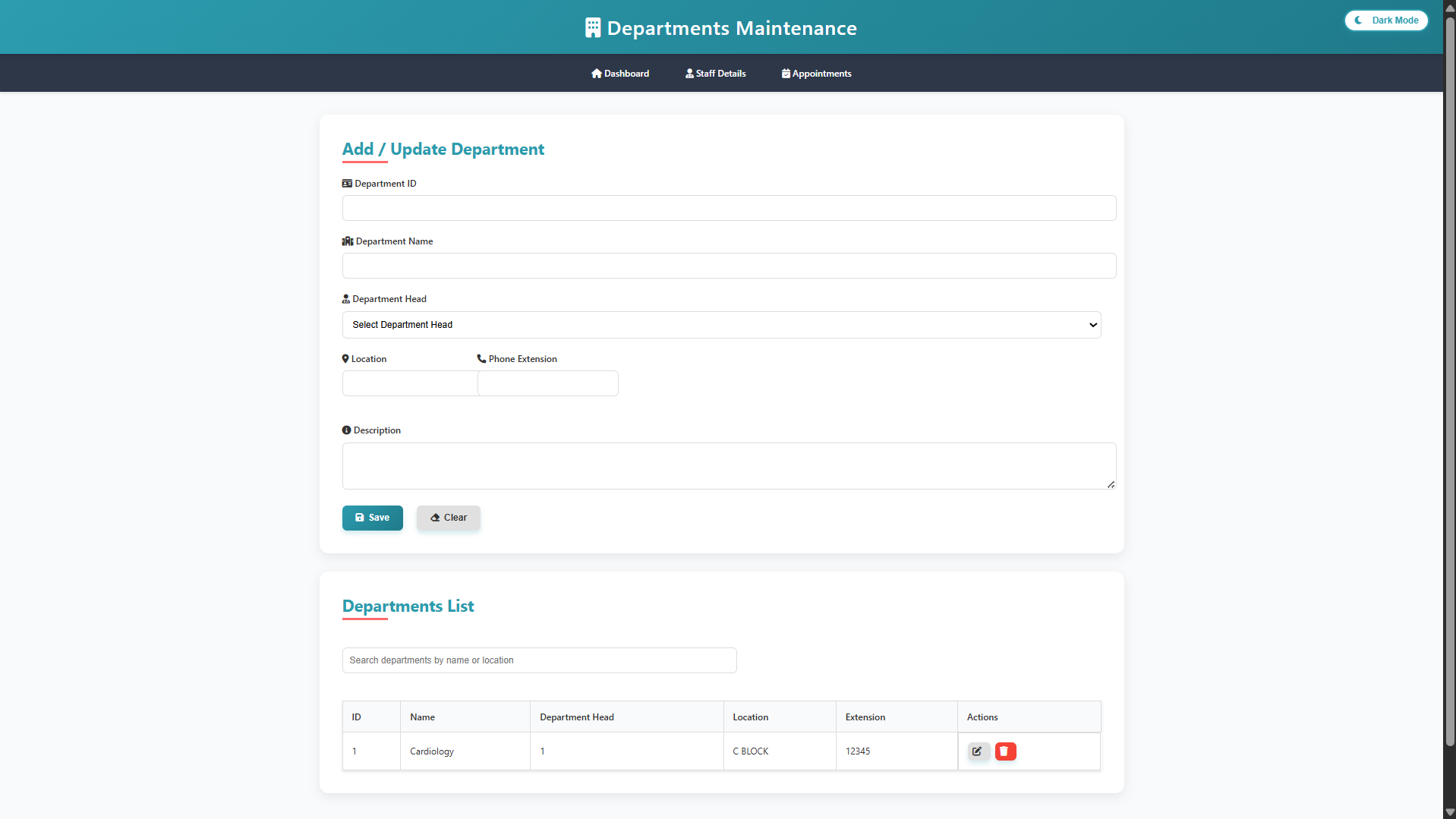
**Figure 3.8: Patient page**

****

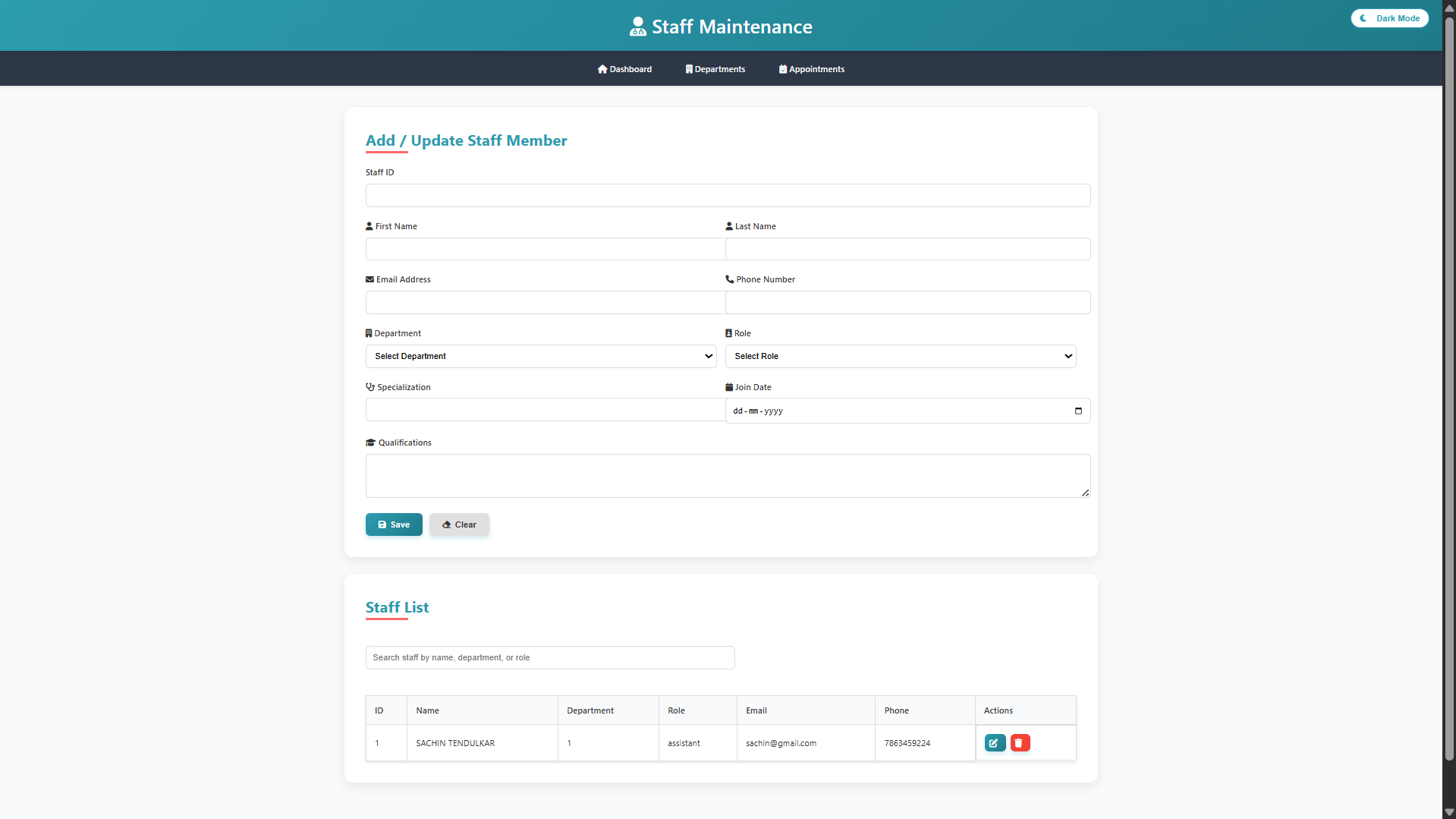
**Figure 3.9: Doctors page**



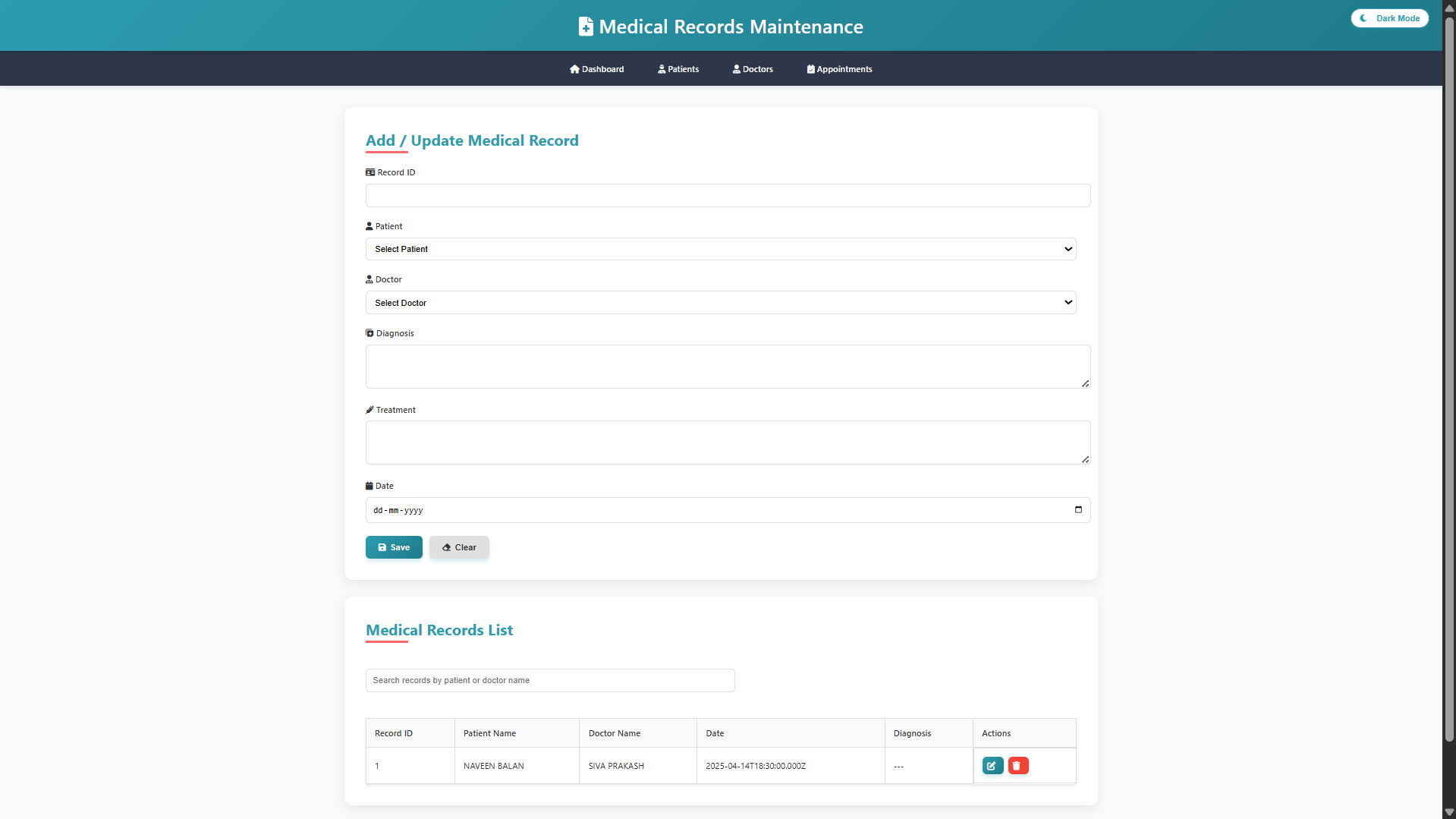
**Figure 3.10: Appointments page**



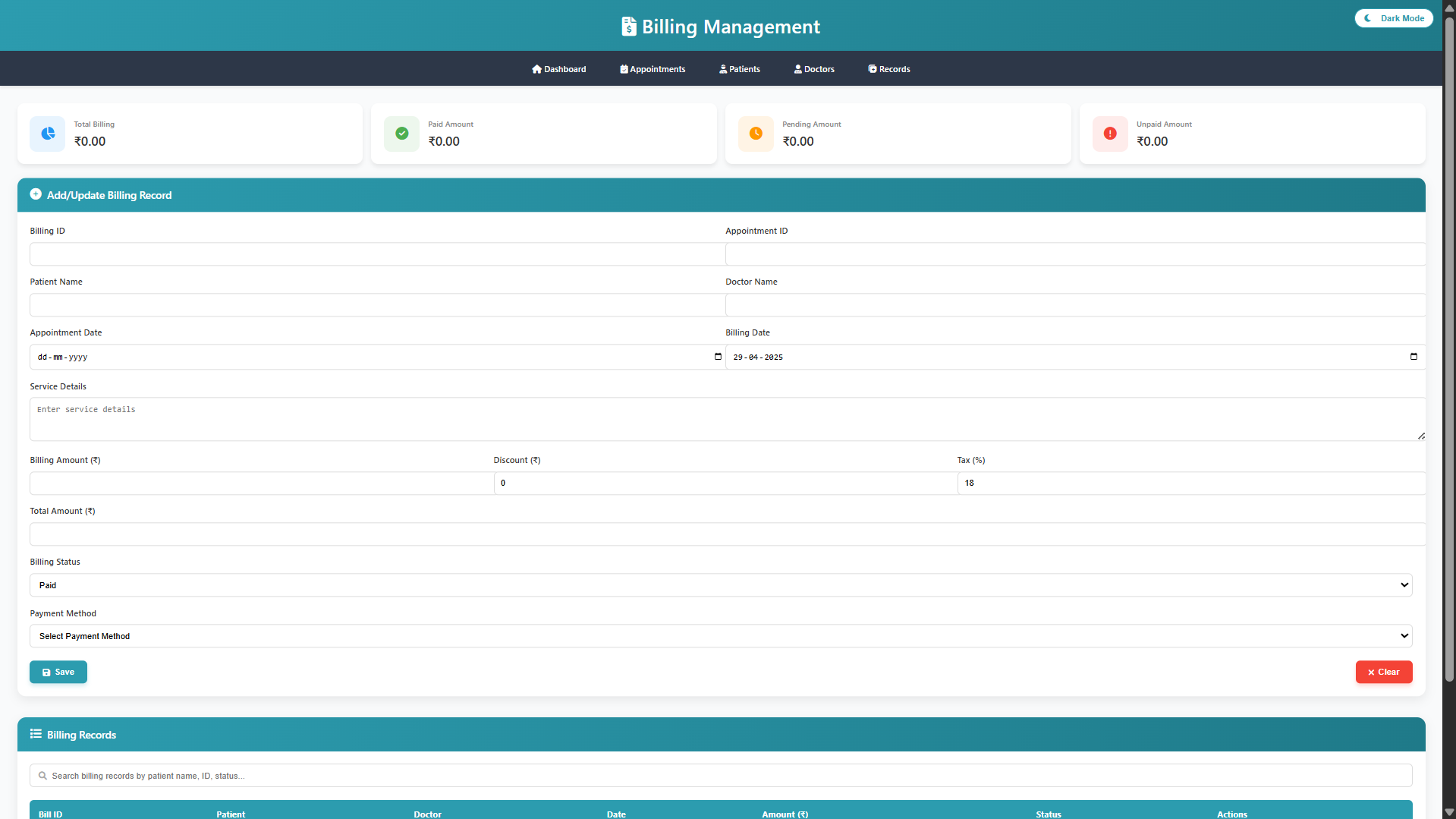
**Figure 3.11: Departments page**



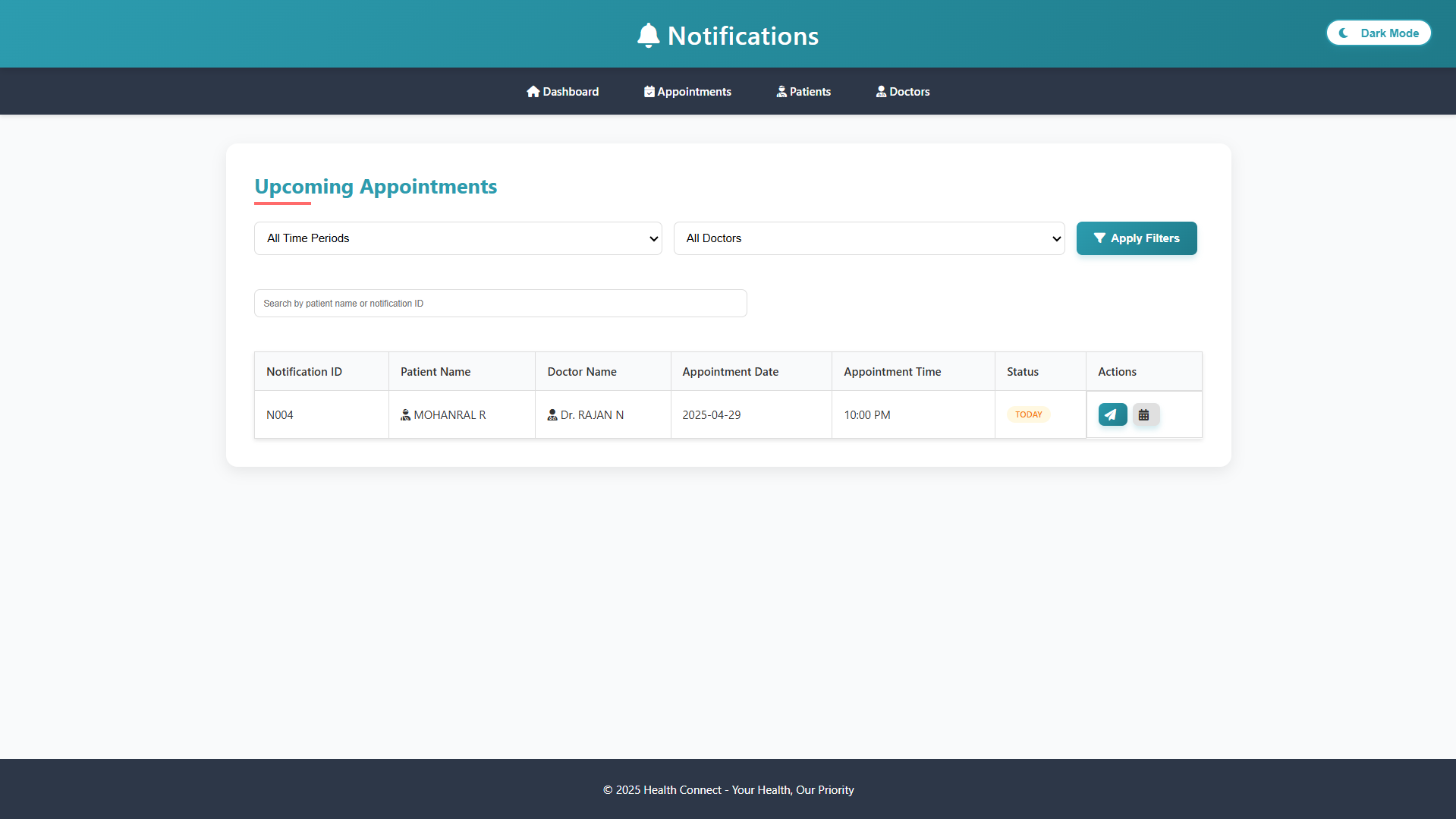
**Figure 3.12: Staffs page**



**Figure 3.13: Medical records page**



**Figure 3.14: Billing Page**



**Figure 3.15: Notification Page**

**CHAPTER 4**

**SYSTEM IMPLEMENTATION**

**4.1LOGIN IMPLEMENTATION**

The login credentials are obtained. If the credentials are OK, then the user is redirected to the homepage

GET username, password

IF username, password valid

RETURN homepage

ELSE

TOAST Invalid Credential

**4.2 SIGNUP IMPLEMENTATION**

The form fields are obtained. If they are valid, then the user is added to the database.

GET requestFields

IF requestFields valid

RETURN added to db

ELSE

TOAST enter valid details

**4.3 APPOINTMENT IMPLEMENTATION**

The form fields are obtained. If they are valid, then the appointment is added to database.

GET requestFields

IF requestFields valid

RETURN added to db

ELSE

TOAST enter valid details

**4.4 PATIENT IMPLEMENTATION**

The form fields are obtained. If they are valid, then the patient is added to database.

GET requestFields

IF requestFields valid

RETURN added to db

ELSE

TOAST enter valid details

**4.5 DOCTOR IMPLEMENTATION**

The form fields are obtained. If they are valid, then the doctor is added to database.

GET requestFields

IF requestFields valid

RETURN added to db

ELSE

TOAST enter valid details

**4.6 DEPARTMENTS IMPLEMENTATION**

The form fields are obtained. If they are valid, then the department is added to database.

GET requestFields

IF requestFields valid

RETURN added to db

ELSE

TOAST enter valid details

**4.7 STAFF IMPLEMENTATION**

The form fields are obtained. If they are valid, then the staff is added to database.

GET requestFields

IF requestFields valid

RETURN added to db

ELSE

TOAST enter valid details

**4.8 MEDICAL RECORDS IMPLEMENTATION**

The form fields are obtained. If they are valid, then the record is added to database.

GET requestFields

IF requestFields valid

RETURN added to db

ELSE

TOAST enter valid details

**4.9 BILLING IMPLEMENTATION**

The form fields are obtained. If they are valid, then the bill is added to database.

GET requestFields

IF requestFields valid

RETURN added to db

ELSE

TOAST enter valid details

**4.10 NOTIFICATION IMPLEMENTATION**

The requestFields required to display appointments is retrieved from the database.

GET requestFields

IF requestFields valid

DISPLAY in the Interface

ELSE

TOAST enter valid details

**4.11 REPORT IMPLEMENTATION**

The requestFields required to display report is retrieved from the database.

GET requestFields

IF requestFields valid

DISPLAY report in the interface

ELSE

TOAST enter valid details

**CHAPTER 5**

**RESULTS AND DISCUSSION**

**5.1 TEST CASES AND RESULTS**

**5.1.1 Test Cases and Results for Login function:**

The Table 5.1, Table 5.2 shows that the possible test data for the both positive and negative test case given below, if the user is already having an account then the output is true otherwise false.

**Table 5.1: Positive Test Case and result for Login**

|  |  |
| --- | --- |
| **Test Case ID** | TC1 |
| **Test Case Description** | It tests whether the given login details are valid or not |
| **Test Data** | user, pass |
| **Expected Output** | TRUE |
| **Result** | PASS |

**Table 5.2: Negative Test Case and result for Login**

|  |  |
| --- | --- |
| **Test Case ID** | TC2 |
| **Test Case Description** | It tests whether the given login details are valid or not |
| **Test Data** | User\_@01 |
| **Expected Output** | FALSE |
| **Result** | PASS |

**5.1.2 Test Cases and Results for Signup function:**

The Table 5.3, Table 5.4 shows that the possible test data for the both positive and negative test case given below.

**Table 5.3: Positive Test Case and result for Signup**

|  |  |
| --- | --- |
| **Test Case ID** | TC3 |
| **Test Case Description** | It tests whether the given signup details are valid or not |
| **Test Data** | user, [user@gmail.com](mailto:user@gmail.com), pass, USER |
| **Expected Output** | Details added to database |
| **Result** | PASS |

**Table 5.4: Negative Test Case and result for Signup**

|  |  |
| --- | --- |
| **Test Case ID** | TC4 |
| **Test Case Description** | It tests whether the given signup details are valid or not |
| **Test Data** | User\_@01 |
| **Expected Output** | Error raised |
| **Result** | PASS |

**5.1.3 Test Cases and Results for Appointment booking function:**

The Table 5.5, Table 5.6 shows that the possible test data for the both positive and negative test case given below.

**Table 5.5: Positive Test Case and result for Appointment booking**

|  |  |
| --- | --- |
| **Test Case ID** | TC5 |
| **Test Case Description** | It tests whether the given appointment booking details are valid or not |
| **Test Data** | MOHANRAL, [mohan@gmail.com](mailto:mohan@gmail.com), 9892791271, General Medicine, Dr. Siva Prakash, 30-04-2025, 22:00 |
| **Expected Output** | Added to database. |
| **Result** | PASS |

**Table 5.6: Negative Test Case and result for Appointment Booking**

|  |  |
| --- | --- |
| **Test Case ID** | TC6 |
| **Test Case Description** | It tests whether the given appointment booking details are valid or not |
| **Test Data** | User\_@01 |
| **Expected Output** | Error raised. |
| **Result** | PASS |

**5.1.4 Test Cases and Results for Patient registration:**

The Table 5.7, Table 5.8 shows that the possible test data for the both positive and negative test case given below.

**Table 5.7: Positive Test Case and result for patient registration**

|  |  |
| --- | --- |
| **Test Case ID** | TC7 |
| **Test Case Description** | It tests whether the given patient details are valid or not |
| **Test Data** | NAVEEN, BALAN, [naveen@gmail.com](mailto:naveen@gmail.com), 9812672636, 21-04-2025, Male, B+ve, IN2891273, Chellapillayarkulam, Tenkasi. |
| **Expected Output** | Added to database. |
| **Result** | PASS |

**Table 5.8: Negative Test Case and result for patient registration**

|  |  |
| --- | --- |
| **Test Case ID** | TC8 |
| **Test Case Description** | It tests whether the given patient details are valid or not |
| **Test Data** | User\_@01 |
| **Expected Output** | Error raised. |
| **Result** | PASS |

**5.1.5 Test Cases and Results for Doctor registration:**

The Table 5.9, Table 5.10 shows that the possible test data for the both positive and negative test case given below.

**Table 5.9: Positive Test Case and result for doctor registration**

|  |  |
| --- | --- |
| **Test Case ID** | TC9 |
| **Test Case Description** | It tests whether the given doctor details are valid or not |
| **Test Data** | SIVA, PRAKASH, [siva@gmail.com](mailto:siva@gmail.com), 8916249823, Cardiology, Surgent, L2897, 21-03-2013, 200, active, MBBS.MS |
| **Expected Output** | Added to database. |
| **Result** | PASS |

**Table 5.10: Negative Test Case and result for doctor** **registration**

|  |  |
| --- | --- |
| **Test Case ID** | TC10 |
| **Test Case Description** | It tests whether the given doctor details are valid or not |
| **Test Data** | User\_@01 |
| **Expected Output** | Error raised. |
| **Result** | PASS |

**CHAPTER 6**

**CONCLUSION AND FUTURE ENHANCEMENTS**

The Medical Appointment Booking System is designed with the primary goal of simplifying the process of scheduling and managing appointments. It offers an efficient and user-friendly platform tailored to meet the needs of both patients and healthcare providers. By automating critical tasks such as appointment booking, notifications, and record-keeping, the system effectively reduces the administrative workload placed on staff, freeing up their time for other essential responsibilities. Simultaneously, it enhances the overall patient experience by providing a seamless and convenient way to secure appointments and stay informed. Looking forward, the system is poised for robust future enhancements aimed at expanding its functionality and adaptability to meet the evolving demands of the healthcare industry. Planned updates include integrating telemedicine capabilities to facilitate remote consultations, providing advanced analytics for data-driven decision-making, and incorporating AI-driven scheduling to optimize resource allocation and reduce wait times. Additionally, the system will offer multilingual support to cater to diverse patient demographics and a mobile application to ensure accessibility on-the-go.

Other enhancements include the integration of wearable devices, allowing for real-time health monitoring and data sharing, as well as expanded payment options to accommodate various financial systems. Automated follow-ups will ensure patients never miss important appointments or check-ups, while emergency appointment handling will provide a quick and efficient response to urgent healthcare needs. Furthermore, interoperability with hospital systems will enable seamless communication and coordination across different healthcare platforms, ensuring a unified experience for both patients and providers.

These forward-thinking updates aim to transform the Medical Appointment Booking System into a comprehensive and adaptive solution, making it a cornerstone technology for modern healthcare. By staying ahead of industry trends, the system aspires to not only meet current needs but also anticipate and address future healthcare challenges, thereby delivering unparalleled value to its users.

**APPENDIX – A**

**SYSTEM REQUIREMENTS**

**HARDWARE REQUIREMENTS:**

|  |  |
| --- | --- |
| Processor | Dual-core processor or higher |
| RAM | 4 GB or more |
| HDD | At least 10 GB of free disk space |

**SOFTWARE REQUIREMENTS:**

|  |  |
| --- | --- |
| Operating System | Windows, macOS, or Linux |
| DBMS | My SQL |
| IDE used | Android Studio |
| JDK Version | 5.1 and above |

**APPENDIX – B**

**SOURCE CODE**

**Index.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Hospital Home Page</title>

    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

    <style>

         :root {

            --primary: #2c9caf;

            --primary-dark: #1f7a89;

            --secondary: #ff6b6b;

            --text-dark: #333;

            --light-bg: #f9fafb;

            --card-bg: #ffffff;

            --success: #4CAF50;

            --error: #f44336;

        }

        [data-theme="dark"] {

            --primary: #3fb8cc;

            --primary-dark: #2c9caf;

            --text-dark: #e0e0e0;

            --text-light: #ffffff;

            --light-bg: #121212;

            --card-bg: #1e1e1e;

            --border-color: #404040;

            --hover-bg: rgba(44, 156, 175, 0.1);

        }

        [data-theme="dark"] body {

            background-color: var(--light-bg);

            color: var(--text-light);

        }

        [data-theme="dark"] .card {

            background-color: var(--card-bg);

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.3);

        }

        [data-theme="dark"] .card p {

            color: #a0aec0;

        }

        [data-theme="dark"] nav {

            background-color: var(--card-bg);

            border-bottom: 1px solid var(--border-color);

        }

        [data-theme="dark"] nav a {

            color: var(--text-light);

        }

        [data-theme="dark"] footer {

            background-color: var(--card-bg);

            border-top: 1px solid var(--border-color);

        }

        body {

            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

            background-color: var(--light-bg);

            margin: 0;

            padding: 0;

            display: flex;

            flex-direction: column;

            min-height: 100vh;

            color: var(--text-dark);

        }

        header {

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            padding: 1.5rem;

            text-align: center;

            position: relative;

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

        }

        header h1 {

            margin: 0;

            font-size: 2.2rem;

            font-weight: 600;

            letter-spacing: 0.5px;

        }

        header .login-btn {

            position: absolute;

            top: 1rem;

            left: 1rem;

            background-color: white;

            color: var(--primary);

            border: 2px solid var(--primary);

            padding: 0.5rem 1rem;

            border-radius: 4px;

            cursor: pointer;

            text-decoration: none;

            font-weight: bold;

            transition: all 0.3s ease;

        }

        header .login-btn:hover {

            background-color: var(--primary-dark);

            color: white;

        }

        nav {

            background-color: #2d3748;

            display: flex;

            justify-content: center;

            padding: 0.7rem;

            box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

        }

        nav a {

            color: white;

            text-decoration: none;

            padding: 0.7rem 1.5rem;

            margin: 0 0.5rem;

            border-radius: 4px;

            transition: all 0.3s ease;

            font-weight: 500;

        }

        nav a:hover {

            background-color: rgba(255, 255, 255, 0.1);

            transform: translateY(-2px);

        }

        .container {

            display: flex;

            flex-direction: column;

            flex: 1;

            align-items: center;

            padding: 2rem;

        }

        .hero {

            background-image: url('Bg.jpg');

            background-size: cover;

            background-position: center;

            width: 100%;

            height: 300px;

            display: flex;

            justify-content: center;

            align-items: center;

            color: white;

            text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);

            border-radius: 10px;

        }

        .hero h1 {

            font-size: 3rem;

        }

        .cards {

            display: grid;

            grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

            gap: 1.5rem;

            width: 100%;

            margin-top: 2rem;

        }

        .card {

            background-color: var(--card-bg);

            border-radius: 10px;

            padding: 1.5rem;

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.05);

            text-align: center;

            transition: transform 0.3s ease, box-shadow 0.3s ease;

        }

        .card:hover {

            transform: translateY(-5px);

            box-shadow: 0 10px 15px rgba(0, 0, 0, 0.1);

        }

        .card img {

            width: 60%;

            border-radius: 10px;

        }

        .card h3 {

            margin-top: 1rem;

            color: var(--primary);

        }

        .card p {

            color: #6b7280;

        }

        .card a button {

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            border: none;

            padding: 0.5rem 1rem;

            border-radius: 4px;

            cursor: pointer;

            margin-top: 1rem;

            transition: all 0.3s ease;

        }

        .card a button:hover {

            background: var(--primary-dark);

            transform: translateY(-2px);

        }

        footer {

            background-color: #2d3748;

            color: white;

            text-align: center;

            padding: 1rem;

            margin-top: auto;

        }

        @media (max-width: 768px) {

            .hero h1 {

                font-size: 2rem;

            }

            .cards {

                grid-template-columns: 1fr;

            }

            .container {

                padding: 1rem;

            }

        }

        /\* Add theme toggle button styles \*/

        .theme-toggle {

            position: fixed;

            top: 1rem;

            right: 1rem;

            z-index: 1000;

            padding: 0.5rem 1rem;

            border-radius: 20px;

            border: 2px solid var(--primary);

            background: var(--card-bg);

            color: var(--primary);

            cursor: pointer;

            display: flex;

            align-items: center;

            gap: 0.5rem;

            transition: all 0.3s ease;

        }

        .theme-toggle:hover {

            background: var(--primary);

            color: var(--text-light);

        }

    </style>

</head>

<body>

    <button class="theme-toggle" onclick="toggleTheme()">

        <i class="fas fa-moon"></i> Dark Mode

    </button>

    <header>

        <a href="login.html" class="login-btn">Login</a>

        <h1><i class="fas fa-hospital-alt"></i> Welcome to Our Hospital</h1>

    </header>

    <nav>

        <a href="about-us.html"><i class="fas fa-info-circle"></i> About Us</a>

        <a href="services.html"><i class="fas fa-concierge-bell"></i> Services</a>

        <a href="contact.html"><i class="fas fa-phone"></i> Contact</a>

    </nav>

    <div class="container">

        <div class="hero">

            <h1>Quality Care for Everyone</h1>

        </div>

        <div class="cards">

            <div class="card">

                <img src="download.jpg" alt="Doctor">

                <h3>Our Doctors</h3>

                <p>Meet our experienced and compassionate medical staff.</p>

                <a href="doctors\_profile.html"><button>Learn More</button></a>

            </div>

            <div class="card">

                <img src="Hospital.jpg" alt="Facility">

                <h3>Our Facilities</h3>

                <p>State-of-the-art facilities to provide the best care.</p>

                <a href="facilities.html"><button>Learn More</button></a>

            </div>

            <div class="card">

                <img src="download (1).jpg" alt="Services">

                <h3>Our Services</h3>

                <p>Comprehensive services to meet all your health needs.</p>

                <a href="services.html"><button>Learn More</button></a>

            </div>

        </div>

    </div>

    <footer>

        <p>&copy; 2025 Our Hospital - Committed to Your Health</p>

    </footer>

    <script>

        function toggleTheme() {

            const html = document.documentElement;

            const currentTheme = html.getAttribute('data-theme');

            const newTheme = currentTheme === 'dark' ? 'light' : 'dark';

            html.setAttribute('data-theme', newTheme);

            localStorage.setItem('theme', newTheme);

            const themeToggle = document.querySelector('.theme-toggle');

            const icon = themeToggle.querySelector('i');

            icon.className = `fas ${newTheme === 'dark' ? 'fa-sun' : 'fa-moon'}`;

            themeToggle.innerHTML = `${icon.outerHTML} ${newTheme === 'dark' ? 'Light Mode' : 'Dark Mode'}`;

        }

        document.addEventListener('DOMContentLoaded', function() {

            const savedTheme = localStorage.getItem('theme') || 'light';

            document.documentElement.setAttribute('data-theme', savedTheme);

            const themeToggle = document.querySelector('.theme-toggle');

            const icon = themeToggle.querySelector('i');

            icon.className = `fas ${savedTheme === 'dark' ? 'fa-sun' : 'fa-moon'}`;

            themeToggle.innerHTML = `${icon.outerHTML} ${savedTheme === 'dark' ? 'Light Mode' : 'Dark Mode'}`;

        });

    </script>

</body>

</html>

**Login.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Login Page - Health Connect</title>

    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

    <style>

         :root {

            --primary: #2c9caf;

            --primary-dark: #1f7a89;

            --secondary: #ff6b6b;

            --text-dark: #333;

            --light-bg: #f9fafb;

            --card-bg: #ffffff;

        }

        body {

            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

            background-color: var(--light-bg);

            display: flex;

            justify-content: center;

            align-items: center;

            height: 100vh;

            margin: 0;

            position: relative;

        }

        .container {

            background-color: var(--card-bg);

            padding: 2rem;

            border-radius: 10px;

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

            text-align: center;

            width: 300px;

            animation: fadeIn 1s ease-in-out;

        }

        h1 {

            margin-bottom: 1.5rem;

            color: var(--primary);

            font-size: 1.8rem;

        }

        .input-group {

            display: flex;

            flex-direction: column;

            margin-bottom: 1rem;

            text-align: left;

        }

        .input-group label {

            margin-bottom: 0.5rem;

            color: var(--text-dark);

            font-weight: 500;

        }

        .input-group input {

            padding: 0.75rem;

            border: 1px solid #ccc;

            border-radius: 4px;

            box-sizing: border-box;

            font-size: 1rem;

        }

        button {

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            padding: 0.75rem;

            border: none;

            border-radius: 4px;

            cursor: pointer;

            width: 100%;

            font-size: 1rem;

            font-weight: bold;

            transition: all 0.3s ease;

        }

        button:hover {

            background: var(--primary-dark);

            transform: translateY(-2px);

        }

        .signup-message {

            margin-top: 1rem;

            color: var(--text-dark);

            font-size: 0.9rem;

        }

        .signup-message a {

            color: var(--primary);

            text-decoration: none;

            font-weight: 500;

        }

        .signup-message a:hover {

            text-decoration: underline;

        }

        @keyframes fadeIn {

            from {

                opacity: 0;

            }

            to {

                opacity: 1;

            }

        }

        .title {

            position: absolute;

            top: 10px;

            left: 20px;

            font-size: 2rem;

            color: var(--primary);

            font-family: 'Brush Script MT', cursive;

            animation: slideIn 1s ease-in-out;

            text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.2);

        }

        @keyframes slideIn {

            from {

                transform: translateX(-100%);

            }

            to {

                transform: translateX(0);

            }

        }

        .background-image {

            position: absolute;

            top: 0;

            left: 0;

            width: 100%;

            height: 100%;

            background-image: url('background.jpg');

            background-size: cover;

            background-position: center;

            opacity: 0.3;

            z-index: -1;

        }

        .home-btn {

            position: absolute;

            top: 10px;

            right: 10px;

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            padding: 0.5rem 1rem;

            border: none;

            border-radius: 4px;

            cursor: pointer;

            text-decoration: none;

            font-weight: bold;

            transition: all 0.3s ease;

        }

        .home-btn:hover {

            background: var(--primary-dark);

            transform: translateY(-2px);

        }

    </style>

</head>

<body>

    <div class="background-image"></div>

    <div class="title">ABS</div>

    <a href="home.html" class="home-btn"><i class="fas fa-home"></i> Home</a>

    <div class="container">

        <h1>Login</h1>

        <form id="loginForm">

            <div class="input-group">

                <label for="username">Username</label>

                <input type="text" id="username" name="username" placeholder="Enter your username" required>

            </div>

            <div class="input-group">

                <label for="password">Password</label>

                <input type="password" id="password" name="password" placeholder="Enter your password" required>

            </div>

            <button type="submit">Login</button>

        </form>

        <p class="signup-message">Don't have an account? <a href="signup.html" id="signupButton">Sign Up</a></p>

    </div>

    <script>

        document.getElementById('loginForm').addEventListener('submit', function(event) {

            event.preventDefault();

            const username = document.getElementById('username').value;

            const password = document.getElementById('password').value;

            fetch('/login', {

                    method: 'POST',

                    headers: {

                        'Content-Type': 'application/json'

                    },

                    body: JSON.stringify({

                        username,

                        password

                    })

                })

                .then(response => response.json())

                .then(data => {

                    if (data.success) {

                        alert('Redirecting to your dashboard...');

                        window.location.href = 'Dashboard.html';

                    } else {

                        alert('Invalid username or password');

                    }

                })

                .catch(error => {

                    console.error('Error:', error);

                    alert('An error occurred. Please try again later.');

                });

        });

        document.getElementById('signupButton').addEventListener('click', function(event) {

            event.preventDefault();

            alert('Redirecting to sign-up page...');

            window.location.href = 'signup.html';

        });

    </script>

</body>

</html>

**Signup.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Sign Up Page - Health Connect</title>

    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

    <style>

         :root {

            --primary: #2c9caf;

            --primary-dark: #1f7a89;

            --secondary: #ff6b6b;

            --text-dark: #333;

            --light-bg: #f9fafb;

            --card-bg: #ffffff;

        }

        body {

            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

            background-color: var(--light-bg);

            display: flex;

            justify-content: center;

            align-items: center;

            height: 100vh;

            margin: 0;

            position: relative;

        }

        .container {

            background-color: var(--card-bg);

            padding: 2rem;

            border-radius: 10px;

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

            text-align: center;

            width: 300px;

            animation: fadeIn 1s ease-in-out;

        }

        h1 {

            margin-bottom: 1.5rem;

            color: var(--primary);

            font-size: 1.8rem;

        }

        .input-group {

            display: flex;

            flex-direction: column;

            margin-bottom: 1rem;

            text-align: left;

        }

        .input-group label {

            margin-bottom: 0.5rem;

            color: var(--text-dark);

            font-weight: 500;

        }

        .input-group input {

            padding: 0.75rem;

            border: 1px solid #ccc;

            border-radius: 4px;

            box-sizing: border-box;

            font-size: 1rem;

        }

        button {

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            padding: 0.75rem;

            border: none;

            border-radius: 4px;

            cursor: pointer;

            width: 100%;

            font-size: 1rem;

            font-weight: bold;

            transition: all 0.3s ease;

        }

        button:hover {

            background: var(--primary-dark);

            transform: translateY(-2px);

        }

        .login-message {

            margin-top: 1rem;

            color: var(--text-dark);

            font-size: 0.9rem;

        }

        .login-message a {

            color: var(--primary);

            text-decoration: none;

            font-weight: 500;

        }

        .login-message a:hover {

            text-decoration: underline;

        }

        @keyframes fadeIn {

            from {

                opacity: 0;

            }

            to {

                opacity: 1;

            }

        }

        .title {

            position: absolute;

            top: 10px;

            left: 20px;

            font-size: 2rem;

            color: var(--primary);

            font-family: 'Brush Script MT', cursive;

            animation: slideIn 1s ease-in-out;

            text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.2);

        }

        @keyframes slideIn {

            from {

                transform: translateX(-100%);

            }

            to {

                transform: translateX(0);

            }

        }

        .background-image {

            position: absolute;

            top: 0;

            left: 0;

            width: 100%;

            height: 100%;

            background-image: url('background.jpg');

            background-size: cover;

            background-position: center;

            opacity: 0.3;

            z-index: -1;

        }

    </style>

</head>

<body>

    <div class="background-image"></div>

    <div class="title">ABS</div>

    <div class="container">

        <h1>Sign Up</h1>

        <form id="signupForm">

            <div class="input-group">

                <label for="username">Username</label>

                <input type="text" id="username" name="username" placeholder="Enter your username" required>

            </div>

            <div class="input-group">

                <label for="email">Email</label>

                <input type="email" id="email" name="email" placeholder="Enter your email" required>

            </div>

            <div class="input-group">

                <label for="password">Password</label>

                <input type="password" id="password" name="password" placeholder="Enter your password" required>

            </div>

            <div class="input-group">

                <label for="name">Name</label>

                <input type="text" id="name" name="name" placeholder="Enter your full name" required>

            </div>

            <button type="submit">Sign Up</button>

        </form>

        <p class="login-message">Already have an account? <a href="login.html">Login</a></p>

    </div>

    <script>

        document.getElementById('signupForm').addEventListener('submit', function(event) {

            event.preventDefault();

            const username = document.getElementById('username').value;

            const email = document.getElementById('email').value;

            const password = document.getElementById('password').value;

            const name = document.getElementById('name').value;

            fetch('/signup', {

                    method: 'POST',

                    headers: {

                        'Content-Type': 'application/json'

                    },

                    body: JSON.stringify({

                        username,

                        email,

                        password,

                        name

                    })

                })

                .then(response => response.json())

                .then(data => {

                    if (data.success) {

                        alert('Sign-up successful!');

                        window.location.href = 'index.html';

                    } else {

                        alert('Sign-up failed: ' + data.message);

                    }

                })

                .catch(error => {

                    console.error('Error:', error);

                    alert('An error occurred. Please try again later.');

                });

        });

    </script>

</body>

</html>

**Dashboard.html**

<!DOCTYPE html>

<html lang="en" data-theme="light">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Dashboard - Health Connect</title>

    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

    <style>

         :root {

            --primary: #2c9caf;

            --primary-dark: #1f7a89;

            --text-dark: #333;

            --text-light: #ffffff;

            --light-bg: #f9fafb;

            --card-bg: #ffffff;

            --success: #4CAF50;

            --error: #f44336;

            --border-color: #ddd;

        }

        [data-theme="dark"] {

            --primary: #3fb8cc;

            --primary-dark: #2c9caf;

            --text-dark: #e0e0e0;

            --text-light: #ffffff;

            --light-bg: #121212;

            --card-bg: #1e1e1e;

            --border-color: #404040;

        }

        body {

            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

            background-color: var(--light-bg);

            margin: 0;

            padding: 0;

            display: flex;

            flex-direction: column;

            min-height: 100vh;

            color: var(--text-dark);

        }

        header {

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            padding: 1.5rem;

            text-align: center;

            position: relative;

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

        }

        header h1 {

            margin: 0;

            font-size: 2.2rem;

            font-weight: 600;

            letter-spacing: 0.5px;

        }

        nav {

            background-color: #2d3748;

            display: flex;

            justify-content: center;

            padding: 0.7rem;

            box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

        }

        nav a {

            color: white;

            text-decoration: none;

            padding: 0.7rem 1.5rem;

            margin: 0 0.5rem;

            border-radius: 4px;

            transition: all 0.3s ease;

            font-weight: 500;

        }

        nav a:hover {

            background-color: rgba(255, 255, 255, 0.1);

            transform: translateY(-2px);

        }

        .container {

            display: flex;

            flex: 1;

        }

        aside {

            background: linear-gradient(to bottom, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            padding: 1.5rem;

            width: 250px;

            box-shadow: 2px 0 5px rgba(0, 0, 0, 0.1);

        }

        aside h2 {

            margin-top: 0;

            font-size: 1.5rem;

            margin-bottom: 1.5rem;

            position: relative;

            padding-bottom: 0.5rem;

        }

        aside h2::after {

            content: '';

            position: absolute;

            left: 0;

            bottom: 0;

            height: 3px;

            width: 40px;

            background-color: var(--secondary);

        }

        aside ul {

            list-style: none;

            padding: 0;

            margin: 0;

        }

        aside ul li {

            margin: 0.8rem 0;

            transition: all 0.3s ease;

        }

        aside ul li a {

            color: white;

            text-decoration: none;

            display: flex;

            align-items: center;

            padding: 0.7rem;

            border-radius: 4px;

            transition: all 0.3s ease;

        }

        aside ul li a:hover {

            background-color: rgba(255, 255, 255, 0.1);

            transform: translateX(5px);

        }

        aside ul li a i {

            margin-right: 10px;

            width: 20px;

            text-align: center;

        }

        main {

            flex: 1;

            padding: 2rem;

            background-color: var(--light-bg);

            overflow-y: auto;

        }

        .dashboard-stats {

            display: grid;

            grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));

            gap: 1.5rem;

            margin-bottom: 2rem;

        }

        .stat-card {

            background-color: var(--card-bg);

            border-radius: 10px;

            padding: 1.5rem;

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

            transition: transform 0.3s ease;

        }

        .stat-info h3 {

            font-size: 2rem;

            margin: 0;

            color: var(--primary);

        }

        .loading {

            animation: pulse 1.5s infinite;

        }

        @keyframes pulse {

            0% {

                opacity: 1;

            }

            50% {

                opacity: 0.5;

            }

            100% {

                opacity: 1;

            }

        }

        .entity-manager {

            background-color: var(--card-bg);

            padding: 1.5rem;

            border-radius: 10px;

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.05);

            margin-bottom: 2rem;

            transition: transform 0.3s ease;

        }

        .entity-manager:hover {

            transform: translateY(-5px);

        }

        .entity-manager h3 {

            margin-top: 0;

            color: var(--primary);

            font-size: 1.5rem;

            position: relative;

            padding-bottom: 0.5rem;

            margin-bottom: 1.5rem;

        }

        .entity-manager h3::after {

            content: '';

            position: absolute;

            left: 0;

            bottom: 0;

            height: 3px;

            width: 50px;

            background-color: var(--secondary);

        }

        .entity-manager table {

            width: 100%;

            border-collapse: collapse;

            margin-top: 1rem;

        }

        .loading,

        .no-data,

        .error-message {

            text-align: center;

            padding: 2rem;

            color: var(--text-dark);

            font-style: italic;

        }

        .error-message {

            color: var(--error);

        }

        .loading {

            color: var(--primary);

        }

        td i {

            margin-right: 0.5rem;

            color: var(--primary);

        }

        #appointmentsTableBody tr:hover {

            background-color: var(--hover-bg);

            transition: background-color 0.3s ease;

        }

        .status-badge {

            padding: 0.4rem 1rem;

            border-radius: 20px;

            font-size: 0.875rem;

            font-weight: 500;

            display: inline-flex;

            align-items: center;

            gap: 0.5rem;

        }

        .status-scheduled {

            background-color: var(--primary);

            color: var(--text-light);

        }

        .status-completed {

            background-color: var(--success);

            color: var(--text-light);

        }

        .status-cancelled {

            background-color: var(--error);

            color: var(--text-light);

        }

        .entity-manager th,

        .entity-manager td {

            padding: 0.8rem;

            text-align: left;

            border-bottom: 1px solid var(--border-color);

        }

        .entity-manager th {

            background-color: var(--primary);

            color: white;

            font-weight: 600;

        }

        .entity-manager tr:hover {

            background-color: #f3f4f6;

        }

        footer {

            background-color: #2d3748;

            color: white;

            text-align: center;

            padding: 1rem;

            margin-top: auto;

        }

        @media (max-width: 992px) {

            .container {

                flex-direction: column;

            }

            aside {

                width: 100%;

                margin-bottom: 1rem;

            }

        }

        @media (max-width: 768px) {

            .dashboard-stats {

                grid-template-columns: 1fr;

            }

            main {

                padding: 1rem;

            }

        }

        /\* Add theme toggle button styles \*/

        .theme-toggle {

            position: fixed;

            top: 1rem;

            right: 1rem;

            z-index: 1000;

            padding: 0.5rem 1rem;

            border-radius: 20px;

            border: 2px solid var(--primary);

            background: var(--card-bg);

            color: var(--primary);

            cursor: pointer;

            display: flex;

            align-items: center;

            gap: 0.5rem;

            transition: all 0.3s ease;

        }

        .theme-toggle:hover {

            background: var(--primary);

            color: var(--text-light);

        }

        /\* Add to your existing style section \*/

        .quick-actions {

            display: grid;

            grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));

            gap: 1rem;

            margin-bottom: 2rem;

        }

        .action-btn {

            background: var(--card-bg);

            border: 1px solid var(--border-color);

            padding: 1rem;

            border-radius: 8px;

            cursor: pointer;

            display: flex;

            align-items: center;

            gap: 0.5rem;

            transition: all 0.3s ease;

            color: var(--text-dark);

            font-size: 1rem;

        }

        .action-btn:hover {

            background: var(--primary);

            color: white;

            transform: translateY(-2px);

        }

        .action-btn i {

            font-size: 1.2rem;

        }

        /\* Add notification styles \*/

        .notifications {

            position: fixed;

            top: 1rem;

            right: 1rem;

            z-index: 1000;

        }

        .notification {

            background: var(--card-bg);

            border-left: 4px solid var(--primary);

            padding: 1rem;

            margin-bottom: 0.5rem;

            border-radius: 4px;

            box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

            display: flex;

            align-items: center;

            gap: 0.5rem;

            animation: slideIn 0.3s ease;

        }

        @keyframes slideIn {

            from {

                transform: translateX(100%);

                opacity: 0;

            }

            to {

                transform: translateX(0);

                opacity: 1;

            }

        }

        /\* Add to your existing style section \*/

        .table-actions {

            display: flex;

            justify-content: space-between;

            align-items: center;

            margin-bottom: 1rem;

        }

        .search-box {

            position: relative;

            flex: 1;

            max-width: 300px;

        }

        .search-box i {

            position: absolute;

            left: 1rem;

            top: 50%;

            transform: translateY(-50%);

            color: var(--text-dark);

        }

        .search-box input {

            width: 100%;

            padding: 0.5rem 1rem 0.5rem 2.5rem;

            border: 1px solid var(--border-color);

            border-radius: 20px;

            background: var(--card-bg);

            color: var(--text-dark);

        }

        .refresh-btn {

            background: var(--card-bg);

            border: 1px solid var(--border-color);

            padding: 0.5rem;

            border-radius: 50%;

            cursor: pointer;

            transition: all 0.3s ease;

        }

        .refresh-btn:hover {

            background: var(--primary);

            color: white;

            transform: rotate(180deg);

        }

        /\* Add to your existing style section \*/

        .filter-select {

            padding: 0.5rem;

            border: 1px solid var(--border-color);

            border-radius: 4px;

            background: var(--card-bg);

            color: var(--text-dark);

            margin-right: 1rem;

        }

        .filter-select:hover {

            border-color: var(--primary);

        }

    </style>

</head>

<body>

    <button class="theme-toggle" onclick="toggleTheme()">

        <i class="fas fa-moon"></i> Dark Mode

    </button>

    <header>

        <h1><i class="fas fa-hospital-user"></i> Health Connect</h1>

    </header>

    <nav>

        <a href="Dashboard.html"><i class="fas fa-home"></i> Home</a>

        <a href="appointments.html"><i class="fas fa-calendar-plus"></i> Book Appointment</a>

        <a href="profile.html"><i class="fas fa-user"></i> Profile</a>

        <a href="index.html"><i class="fas fa-sign-out-alt"></i> Log out</a>

    </nav>

    <div class="container">

        <aside>

            <h2>Navigation</h2>

            <ul>

                <li><a href="patients.html"><i class="fas fa-users"></i> Patients</a></li>

                <li><a href="doctors.html"><i class="fas fa-user-md"></i> Doctors</a></li>

                <li><a href="appointments.html"><i class="fas fa-calendar-check"></i> Appointments</a></li>

                <li><a href="departments.html"><i class="fas fa-hospital"></i> Departments</a></li>

                <li><a href="staff.html"><i class="fas fa-user-nurse"></i> Staff</a></li>

                <li><a href="medical\_records.html"><i class="fas fa-notes-medical"></i> Medical Records</a></li>

                <li><a href="billing.html"><i class="fas fa-file-invoice-dollar"></i> Billing</a></li>

                <li><a href="notifications.html"><i class="fas fa-bell"></i> Notifications</a></li>

                <li><a href="reports.html"><i class="fas fa-chart-bar"></i> Reports</a></li>

                <li><a href="settings.html"><i class="fas fa-cog"></i> Settings</a></li>

            </ul>

        </aside>

        <main>

            <div class="dashboard-stats">

                <div class="stat-card">

                    <div class="stat-icon">

                        <i class="fas fa-user-injured"></i>

                    </div>

                    <div class="stat-info">

                        <h3 id="total-patients">...</h3>

                        <p>Total Patients</p>

                    </div>

                </div>

                <div class="stat-card">

                    <div class="stat-icon">

                        <i class="fas fa-user-md"></i>

                    </div>

                    <div class="stat-info">

                        <h3 id="total-doctors">...</h3>

                        <p>Active Doctors</p>

                    </div>

                </div>

                <div class="stat-card">

                    <div class="stat-icon">

                        <i class="fas fa-calendar-check"></i>

                    </div>

                    <div class="stat-info">

                        <h3 id="today-appointments">...</h3>

                        <p>Total Appointments</p>

                    </div>

                </div>

            </div>

            <div class="quick-actions">

                <button onclick="location.href='appointments.html'" class="action-btn">

                    <i class="fas fa-calendar-plus"></i>

                    New Appointment

                </button>

                <button onclick="location.href='patients.html'" class="action-btn">

                    <i class="fas fa-user-plus"></i>

                    Add Patient

                </button>

                <button onclick="location.href='medical\_records.html'" class="action-btn">

                    <i class="fas fa-notes-medical"></i>

                    Add Record

                </button>

                <button onclick="location.href='reports.html'" class="action-btn">

                    <i class="fas fa-chart-line"></i>

                    View Reports

                </button>

            </div>

            <div class="entity-manager">

                <div class="table-actions">

                    <h3><i class="fas fa-calendar-day"></i> Today's Appointments</h3>

                    <div class="search-box">

                        <i class="fas fa-search"></i>

                        <input type="text" id="search-appointments" placeholder="Search appointments...">

                    </div>

                    <button id="refresh-btn" class="refresh-btn" title="Refresh data">

                        <i class="fas fa-sync-alt"></i>

                    </button>

                </div>

                <table id="today-appointments-table">

                    <thead>

                        <tr>

                            <th>Patient Name</th>

                            <th>Doctor</th>

                            <th>Department</th>

                            <th>Date&Time</th>

                            <th>Status</th>

                        </tr>

                    </thead>

                    <tbody id="appointmentsTableBody">

                        <!-- Data will be loaded dynamically -->

                    </tbody>

                </table>

            </div>

        </main>

    </div>

    <footer>

        <p>&copy; 2025 Health Connect - Your Health, Our Priority</p>

    </footer>

    <script>

        function toggleTheme() {

            const html = document.documentElement;

            const currentTheme = html.getAttribute('data-theme');

            const newTheme = currentTheme === 'dark' ? 'light' : 'dark';

            html.setAttribute('data-theme', newTheme);

            localStorage.setItem('theme', newTheme);

            const themeToggle = document.querySelector('.theme-toggle');

            const icon = themeToggle.querySelector('i');

            icon.className = `fas ${newTheme === 'dark' ? 'fa-sun' : 'fa-moon'}`;

            themeToggle.innerHTML = `${icon.outerHTML} ${newTheme === 'dark' ? 'Light Mode' : 'Dark Mode'}`;

        }

        function showNotification(message, type = 'info') {

            const notifications = document.getElementById('notifications');

            const notification = document.createElement('div');

            notification.className = `notification ${type}`;

            notification.innerHTML = `

                <i class="fas fa-${type === 'success' ? 'check' : 'info'}-circle"></i>

                <span>${message}</span>

            `;

            notifications.appendChild(notification);

            setTimeout(() => notification.remove(), 3000);

        }

        function fetchDashboardStats() {

            const patientCount = document.getElementById('total-patients');

            const doctorCount = document.getElementById('total-doctors');

            const appointmentCount = document.getElementById('today-appointments');

            fetch('/api/dashboard/stats')

                .then(response => response.json())

                .then(data => {

                    if (data.success && data.stats) {

                        patientCount.textContent = data.stats.patients;

                        doctorCount.textContent = data.stats.doctors;

                        appointmentCount.textContent = data.stats.todayAppointments;

                    }

                })

                .catch(error => {

                    console.error('Error:', error);

                    patientCount.textContent = '0';

                    doctorCount.textContent = '0';

                    appointmentCount.textContent = '0';

                });

        }

        function fetchTodayAppointments() {

            const tbody = document.getElementById('appointmentsTableBody');

            tbody.innerHTML = '<tr><td colspan="5" class="loading"><i class="fas fa-spinner fa-spin"></i> Loading...</td></tr>';

            fetch('/api/notifications/appointments')

                .then(response => {

                    if (!response.ok) {

                        throw new Error('Failed to fetch appointments');

                    }

                    return response.json();

                })

                .then(appointments => {

                    tbody.innerHTML = '';

                    if (!appointments || appointments.length === 0) {

                        tbody.innerHTML = `

                            <tr><td colspan="5" class="no-data">

                                <i class="fas fa-calendar-xmark"></i> No appointments for today

                            </td></tr>`;

                        document.getElementById('today-appointments').textContent = '0';

                        return;

                    }

                    appointments.forEach(appt => {

                        // Updated date formatting to use appointment\_date

                        const date = appt.appointment\_date ? new Date(appt.appointment\_date).toLocaleDateString('en-GB', {

                            day: '2-digit',

                            month: 'short',

                            year: 'numeric'

                        }) : 'N/A';

                        tbody.innerHTML += `

                            <tr>

                                <td><i class="fas fa-user-injured"></i> ${appt.patient\_name || 'N/A'}</td>

                                <td><i class="fas fa-user-md"></i> ${appt.doctor\_name || 'N/A'}</td>

                                <td><i class="fas fa-hospital"></i> ${appt.department || 'N/A'}</td>

                                <td><i class="fas fa-calendar"></i> ${date}</td>

                                <td><span class="status-badge status-scheduled">Scheduled</span></td>

                            </tr>

                        `;

                    });

                    document.getElementById('today-appointments').textContent = appointments.length;

                })

                .catch(error => {

                    console.error('Error:', error);

                    tbody.innerHTML = `

                        <tr><td colspan="5" class="error-message">

                            <i class="fas fa-exclamation-circle"></i> Error loading appointments

                        </td></tr>`;

                    document.getElementById('today-appointments').textContent = '0';

                });

        }

        // Initialize on page load

        document.addEventListener('DOMContentLoaded', function() {

            const savedTheme = localStorage.getItem('theme') || 'light';

            document.documentElement.setAttribute('data-theme', savedTheme);

            fetchDashboardStats();

            fetchTodayAppointments();

            // Refresh every 5 minutes

            setInterval(() => {

                fetchDashboardStats();

                fetchTodayAppointments();

            }, 300000);

            // Refresh button click handler

            document.getElementById('refresh-btn').addEventListener('click', () => {

                fetchDashboardStats();

                fetchTodayAppointments();

                showNotification('Dashboard updated', 'success');

            });

        });

    </script>

</body>

</html>

**Appointments.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Book an Appointment - Health Connect</title>

    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

    <style>

         :root {

            --primary: #2c9caf;

            --primary-dark: #1f7a89;

            --secondary: #ff6b6b;

            --text-dark: #333;

            --light-bg: #f9fafb;

            --card-bg: #ffffff;

            --success: #4CAF50;

            --error: #f44336;

        }

        body {

            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

            background-color: var(--light-bg);

            margin: 0;

            padding: 0;

            display: flex;

            flex-direction: column;

            min-height: 100vh;

            color: var(--text-dark);

        }

        header {

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            padding: 1.5rem;

            text-align: center;

            position: relative;

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

        }

        header h1 {

            margin: 0;

            font-size: 2.2rem;

            font-weight: 600;

            letter-spacing: 0.5px;

        }

        nav {

            background-color: #2d3748;

            display: flex;

            justify-content: center;

            padding: 0.7rem;

            box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

        }

        nav a {

            color: white;

            text-decoration: none;

            padding: 0.7rem 1.5rem;

            margin: 0 0.5rem;

            border-radius: 4px;

            transition: all 0.3s ease;

            font-weight: 500;

        }

        nav a:hover {

            background-color: rgba(255, 255, 255, 0.1);

            transform: translateY(-2px);

        }

        .container {

            flex: 1;

            padding: 2.5rem;

            display: flex;

            flex-direction: column;

            align-items: center;

        }

        .content {

            width: 100%;

            max-width: 800px;

            background-color: var(--card-bg);

            padding: 2.5rem;

            border-radius: 12px;

            box-shadow: 0 5px 15px rgba(0, 0, 0, 0.08);

            transition: transform 0.3s ease;

        }

        .content:hover {

            transform: translateY(-5px);

        }

        .content h2 {

            color: var(--primary);

            margin-top: 0;

            font-size: 1.8rem;

            position: relative;

            padding-bottom: 0.5rem;

            margin-bottom: 1.5rem;

        }

        .content h2::after {

            content: '';

            position: absolute;

            left: 0;

            bottom: 0;

            height: 3px;

            width: 60px;

            background-color: var(--secondary);

        }

        .content p {

            color: #666;

            margin-bottom: 2rem;

            line-height: 1.6;

        }

        .form-group {

            margin-bottom: 1.5rem;

        }

        label {

            display: block;

            margin-bottom: 0.7rem;

            font-weight: 600;

            color: var(--text-dark);

        }

        input,

        select {

            width: 100%;

            padding: 0.75rem;

            border: 1px solid #ddd;

            border-radius: 6px;

            font-size: 1rem;

            transition: border 0.3s ease, box-shadow 0.3s ease;

        }

        input:focus,

        select:focus {

            outline: none;

            border-color: var(--primary);

            box-shadow: 0 0 0 3px rgba(44, 156, 175, 0.2);

        }

        .input-with-icon {

            position: relative;

        }

        .input-with-icon i {

            position: absolute;

            top: 50%;

            left: 10px;

            transform: translateY(-50%);

            color: #aaa;

        }

        .input-with-icon input,

        .input-with-icon select {

            padding-left: 35px;

        }

        button {

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            border: none;

            padding: 0.8rem 1.5rem;

            border-radius: 6px;

            cursor: pointer;

            font-size: 1rem;

            font-weight: 600;

            transition: all 0.3s ease;

            display: inline-flex;

            align-items: center;

            justify-content: center;

            box-shadow: 0 4px 6px rgba(44, 156, 175, 0.2);

        }

        button i {

            margin-right: 0.5rem;

        }

        button:hover {

            background: linear-gradient(135deg, var(--primary-dark) 0%, var(--primary) 100%);

            transform: translateY(-2px);

            box-shadow: 0 6px 8px rgba(44, 156, 175, 0.3);

        }

        .message {

            padding: 1rem;

            border-radius: 6px;

            margin-top: 1.5rem;

            font-weight: 500;

            display: none;

            animation: fadeIn 0.5s ease;

        }

        .success-message {

            color: #2e7d32;

            background-color: #e8f5e9;

            border-left: 4px solid var(--success);

        }

        .error-message {

            color: #c62828;

            background-color: #ffebee;

            border-left: 4px solid var(--error);

        }

        @keyframes fadeIn {

            from {

                opacity: 0;

                transform: translateY(-10px);

            }

            to {

                opacity: 1;

                transform: translateY(0);

            }

        }

        .form-row {

            display: flex;

            gap: 1rem;

        }

        .form-row .form-group {

            flex: 1;

        }

        footer {

            background-color: #2d3748;

            color: white;

            text-align: center;

            padding: 1rem;

            margin-top: auto;

        }

        /\* Responsive adjustments \*/

        @media (max-width: 768px) {

            .form-row {

                flex-direction: column;

                gap: 0;

            }

            .container {

                padding: 1rem;

            }

            .content {

                padding: 1.5rem;

            }

        }

        /\* Add these dark theme variables after the existing :root variables \*/

        [data-theme="dark"] {

            --primary: #3fb8cc;

            --primary-dark: #2c9caf;

            --text-dark: #e0e0e0;

            --text-light: #ffffff;

            --light-bg: #121212;

            --card-bg: #1e1e1e;

            --border-color: #404040;

            --hover-bg: rgba(44, 156, 175, 0.1);

        }

        /\* Add dark mode styles before the @media query \*/

        [data-theme="dark"] body {

            background-color: var(--light-bg);

            color: var(--text-light);

        }

        [data-theme="dark"] .content {

            background-color: var(--card-bg);

            box-shadow: 0 5px 15px rgba(0, 0, 0, 0.3);

        }

        [data-theme="dark"] .content p {

            color: #a0aec0;

        }

        [data-theme="dark"] input,

        [data-theme="dark"] select,

        [data-theme="dark"] textarea {

            background-color: var(--card-bg);

            color: var(--text-light);

            border-color: var(--border-color);

        }

        [data-theme="dark"] label {

            color: var(--text-light);

        }

        [data-theme="dark"] nav {

            background-color: var(--card-bg);

        }

        [data-theme="dark"] footer {

            background-color: var(--card-bg);

            border-top: 1px solid var(--border-color);

        }

        /\* Add theme toggle button styles \*/

        .theme-toggle {

            position: fixed;

            top: 1rem;

            right: 1rem;

            z-index: 1000;

            padding: 0.5rem 1rem;

            border-radius: 20px;

            border: 2px solid var(--primary);

            background: var(--card-bg);

            color: var(--primary);

            cursor: pointer;

            display: flex;

            align-items: center;

            gap: 0.5rem;

            transition: all 0.3s ease;

        }

        .theme-toggle:hover {

            background: var(--primary);

            color: var(--text-light);

        }

    </style>

</head>

<body>

    <button class="theme-toggle" onclick="toggleTheme()">

        <i class="fas fa-moon"></i> Dark Mode

    </button>

    <header>

        <h1><i class="fas fa-hospital-user"></i> Health Connect</h1>

    </header>

    <nav>

        <a href="Dashboard.html"><i class="fas fa-home"></i> Home</a>

        <a href="profile.html"><i class="fas fa-user"></i> Profile</a>

        <a href="index.html"><i class="fas fa-sign-out-alt"></i> Log out</a>

    </nav>

    <div class="container">

        <div class="content">

            <h2>Schedule Your Appointment</h2>

            <p>Fill out the form below to book an appointment with our specialists. We'll confirm your booking via email and phone.</p>

            <form id="appointmentForm">

                <div class="form-row">

                    <div class="form-group">

                        <label for="name"><i class="fas fa-user"></i> Full Name</label>

                        <input type="text" id="name" name="name" placeholder="Enter your full name" required>

                    </div>

                    <div class="form-group">

                        <label for="email"><i class="fas fa-envelope"></i> Email Address</label>

                        <input type="email" id="email" name="email" placeholder="Enter your email" required>

                    </div>

                </div>

                <div class="form-group">

                    <label for="phone"><i class="fas fa-phone"></i> Phone Number</label>

                    <input type="tel" id="phone" name="phone" placeholder="Enter your phone number" required>

                </div>

                <div class="form-group">

                    <label for="department"><i class="fas fa-hospital"></i> Department</label>

                    <select id="department" name="department" required>

                        <option value="">Select Department</option>

                        <option value="general-medicine">General Medicine</option>

                        <option value="cardiology">Cardiology</option>

                        <option value="pediatrics">Pediatrics</option>

                        <option value="orthopedics">Orthopedics</option>

                        <option value="gynecology">Gynecology</option>

                        <option value="dermatology">Dermatology</option>

                    </select>

                </div>

                <div class="form-group">

                    <label for="doctor"><i class="fas fa-user-md"></i> Doctor</label>

                    <select id="doctor" name="doctor" required>

                        <option value="">Select Doctor</option>

                        <!-- Will be populated dynamically based on department selection -->

                    </select>

                </div>

                <div class="form-row">

                    <div class="form-group">

                        <label for="date"><i class="fas fa-calendar"></i> Preferred Date</label>

                        <input type="date" id="date" name="date" required>

                    </div>

                    <div class="form-group">

                        <label for="time"><i class="fas fa-clock"></i> Preferred Time</label>

                        <input type="time" id="time" name="time" required>

                    </div>

                </div>

                <div class="form-group">

                    <label for="reason"><i class="fas fa-comment-medical"></i> Reason for Visit</label>

                    <textarea id="reason" name="reason" rows="3" placeholder="Briefly describe your symptoms or reason for visit"></textarea>

                </div>

                <button type="submit"><i class="fas fa-calendar-check"></i> Book Appointment</button>

                <div class="message success-message" id="successMessage"><i class="fas fa-check-circle"></i> Your appointment has been booked successfully!</div>

                <div class="message error-message" id="errorMessage"><i class="fas fa-exclamation-circle"></i> There was an error booking your appointment. Please try again.</div>

            </form>

        </div>

    </div>

    <footer>

        <p>&copy; 2025 Health Connect - Your Health, Our Priority</p>

    </footer>

    <script>

        function handleFormSubmission(event) {

            event.preventDefault();

            const formData = new FormData(event.target);

            const appointmentData = {

                name: formData.get('name'),

                email: formData.get('email'),

                phone: formData.get('phone'),

                department: formData.get('department'),

                doctor\_id: formData.get('doctor'),

                date: formData.get('date'),

                time: formData.get('time'),

                reason: formData.get('reason')

            };

            const successMessage = document.getElementById('successMessage');

            const errorMessage = document.getElementById('errorMessage');

            fetch('/api/book-appointment', {

                    method: 'POST',

                    headers: {

                        'Content-Type': 'application/json'

                    },

                    body: JSON.stringify(appointmentData)

                })

                .then(response => response.json())

                .then(data => {

                    if (data.success) {

                        successMessage.style.display = 'block';

                        errorMessage.style.display = 'none';

                        event.target.reset();

                        setTimeout(() => {

                            successMessage.style.display = 'none';

                        }, 5000);

                    } else {

                        throw new Error(data.message || 'Failed to book appointment');

                    }

                })

                .catch(error => {

                    console.error('Error:', error);

                    successMessage.style.display = 'none';

                    errorMessage.style.display = 'block';

                    errorMessage.textContent = error.message || 'There was an error booking your appointment.';

                });

        }

        function loadDoctorsByDepartment(department) {

            const doctorSelect = document.getElementById('doctor');

            doctorSelect.innerHTML = '<option value="">Loading doctors...</option>';

            doctorSelect.disabled = true;

            if (!department) {

                doctorSelect.innerHTML = '<option value="">Select Doctor</option>';

                doctorSelect.disabled = true;

                return;

            }

            fetch(`/api/doctors/by-department?department=${encodeURIComponent(department)}`)

                .then(response => {

                    if (!response.ok) throw new Error('Network response was not ok');

                    return response.json();

                })

                .then(doctors => {

                    doctorSelect.innerHTML = '<option value="">Select Doctor</option>';

                    if (doctors && doctors.length > 0) {

                        doctors.forEach(doctor => {

                            const option = document.createElement('option');

                            option.value = doctor.id;

                            option.textContent = `Dr. ${doctor.first\_name} ${doctor.last\_name} - ${doctor.specialization}`;

                            doctorSelect.appendChild(option);

                        });

                    } else {

                        doctorSelect.innerHTML = '<option value="">No doctors available</option>';

                    }

                })

                .catch(error => {

                    console.error('Error:', error);

                    doctorSelect.innerHTML = '<option value="">Error loading doctors</option>';

                })

                .finally(() => {

                    doctorSelect.disabled = false;

                });

        }

        document.addEventListener('DOMContentLoaded', function() {

            // Set initial theme

            const savedTheme = localStorage.getItem('theme') || 'light';

            document.documentElement.setAttribute('data-theme', savedTheme);

            updateThemeButton(savedTheme);

            // Set minimum date to today

            const dateInput = document.getElementById('date');

            const today = new Date();

            dateInput.min = today.toISOString().split('T')[0];

            // Add event listeners

            document.getElementById('department').addEventListener('change', (e) => {

                loadDoctorsByDepartment(e.target.value);

            });

            document.getElementById('appointmentForm').addEventListener('submit', handleFormSubmission);

        });

        function updateThemeButton(theme) {

            const themeToggle = document.querySelector('.theme-toggle');

            const icon = themeToggle.querySelector('i');

            icon.className = `fas ${theme === 'dark' ? 'fa-sun' : 'fa-moon'}`;

            themeToggle.innerHTML = `${icon.outerHTML} ${theme === 'dark' ? 'Light Mode' : 'Dark Mode'}`;

        }

        function toggleTheme() {

            const html = document.documentElement;

            const currentTheme = html.getAttribute('data-theme');

            const newTheme = currentTheme === 'dark' ? 'light' : 'dark';

            html.setAttribute('data-theme', newTheme);

            localStorage.setItem('theme', newTheme);

            updateThemeButton(newTheme);

        }

    </script>

</body>

</html>

**Medicalrecord.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Medical Records Maintenance - Health Connect</title>

    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

    <style>

         :root {

            --primary: #2c9caf;

            --primary-dark: #1f7a89;

            --secondary: #ff6b6b;

            --text-dark: #333;

            --light-bg: #f9fafb;

            --card-bg: #ffffff;

            --success: #4CAF50;

            --error: #f44336;

        }

        [data-theme="dark"] {

            --primary: #3fb8cc;

            --primary-dark: #2c9caf;

            --text-dark: #e0e0e0;

            --text-light: #ffffff;

            --light-bg: #121212;

            --card-bg: #1e1e1e;

            --border-color: #404040;

            --hover-bg: rgba(44, 156, 175, 0.1);

            --table-header: #2d2d2d;

        }

        [data-theme="dark"] body {

            background-color: var(--light-bg);

            color: var(--text-light);

        }

        [data-theme="dark"] .content {

            background-color: var(--card-bg);

            box-shadow: 0 5px 15px rgba(0, 0, 0, 0.3);

        }

        [data-theme="dark"] input,

        [data-theme="dark"] select,

        [data-theme="dark"] textarea {

            background-color: var(--card-bg);

            color: var(--text-light);

            border-color: var(--border-color);

        }

        [data-theme="dark"] label {

            color: var(--text-light);

        }

        [data-theme="dark"] table {

            border-color: var(--border-color);

        }

        [data-theme="dark"] th {

            background-color: var(--table-header);

            color: var(--text-light);

            border-color: var(--border-color);

        }

        [data-theme="dark"] td {

            border-color: var(--border-color);

            color: var(--text-light);

        }

        [data-theme="dark"] tr:hover {

            background-color: var(--hover-bg);

        }

        [data-theme="dark"] nav {

            background-color: var(--card-bg);

            border-bottom: 1px solid var(--border-color);

        }

        [data-theme="dark"] footer {

            background-color: var(--card-bg);

            border-top: 1px solid var(--border-color);

        }

        body {

            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

            background-color: var(--light-bg);

            margin: 0;

            padding: 0;

            display: flex;

            flex-direction: column;

            min-height: 100vh;

            color: var(--text-dark);

        }

        header {

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            padding: 1.5rem;

            text-align: center;

            position: relative;

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

        }

        header h1 {

            margin: 0;

            font-size: 2.2rem;

            font-weight: 600;

            letter-spacing: 0.5px;

        }

        nav {

            background-color: #2d3748;

            display: flex;

            justify-content: center;

            padding: 0.7rem;

            box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

        }

        nav a {

            color: white;

            text-decoration: none;

            padding: 0.7rem 1.5rem;

            margin: 0 0.5rem;

            border-radius: 4px;

            transition: all 0.3s ease;

            font-weight: 500;

        }

        nav a:hover {

            background-color: rgba(255, 255, 255, 0.1);

            transform: translateY(-2px);

        }

        .container {

            flex: 1;

            padding: 2.5rem;

            display: flex;

            flex-direction: column;

            align-items: center;

        }

        .content {

            width: 100%;

            max-width: 1000px;

            background-color: var(--card-bg);

            padding: 2.5rem;

            border-radius: 12px;

            box-shadow: 0 5px 15px rgba(0, 0, 0, 0.08);

            margin-bottom: 2rem;

        }

        .content h2 {

            color: var(--primary);

            margin-top: 0;

            font-size: 1.8rem;

            position: relative;

            padding-bottom: 0.5rem;

            margin-bottom: 1.5rem;

        }

        .content h2::after {

            content: '';

            position: absolute;

            left: 0;

            bottom: 0;

            height: 3px;

            width: 60px;

            background-color: var(--secondary);

        }

        .form-group {

            margin-bottom: 1.5rem;

        }

        label {

            display: block;

            margin-bottom: 0.7rem;

            font-weight: 600;

            color: var(--text-dark);

        }

        input,

        textarea,

        select {

            width: 100%;

            padding: 0.75rem;

            border: 1px solid #ddd;

            border-radius: 6px;

            font-size: 1rem;

            transition: border 0.3s ease, box-shadow 0.3s ease;

        }

        input:focus,

        textarea:focus,

        select:focus {

            outline: none;

            border-color: var(--primary);

            box-shadow: 0 0 0 3px rgba(44, 156, 175, 0.2);

        }

        button {

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            border: none;

            padding: 0.8rem 1.5rem;

            border-radius: 6px;

            cursor: pointer;

            font-size: 1rem;

            font-weight: 600;

            transition: all 0.3s ease;

            display: inline-flex;

            align-items: center;

            justify-content: center;

            box-shadow: 0 4px 6px rgba(44, 156, 175, 0.2);

            margin-right: 0.5rem;

        }

        button i {

            margin-right: 0.5rem;

        }

        button:hover {

            background: linear-gradient(135deg, var(--primary-dark) 0%, var(--primary) 100%);

            transform: translateY(-2px);

            box-shadow: 0 6px 8px rgba(44, 156, 175, 0.3);

        }

        button.secondary {

            background: #e0e0e0;

            color: #333;

        }

        button.secondary:hover {

            background: #d0d0d0;

        }

        button.delete {

            background: var(--error);

        }

        button.delete:hover {

            background: #d32f2f;

        }

        .form-row {

            display: flex;

            gap: 1rem;

            margin-bottom: 1.5rem;

        }

        .form-buttons {

            display: flex;

            gap: 1rem;

        }

        table {

            width: 100%;

            border-collapse: collapse;

            margin-top: 1.5rem;

            box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

        }

        th,

        td {

            padding: 1rem;

            border: 1px solid #ddd;

            text-align: left;

        }

        th {

            background-color: var(--light-bg);

            font-weight: 600;

        }

        tr:hover {

            background-color: rgba(44, 156, 175, 0.05);

        }

        .action-buttons {

            display: flex;

            gap: 0.5rem;

        }

        .action-buttons button {

            padding: 0.5rem;

            margin: 0;

        }

        .message {

            padding: 1rem;

            border-radius: 6px;

            margin-top: 1.5rem;

            font-weight: 500;

            display: none;

            animation: fadeIn 0.5s ease;

        }

        .success-message {

            color: #2e7d32;

            background-color: #e8f5e9;

            border-left: 4px solid var(--success);

        }

        .error-message {

            color: #c62828;

            background-color: #ffebee;

            border-left: 4px solid var(--error);

        }

        #search-input {

            margin: 1.5rem 0;

            padding: 0.75rem;

            border: 1px solid #ddd;

            border-radius: 6px;

            width: 50%;

        }

        footer {

            background-color: #2d3748;

            color: white;

            text-align: center;

            padding: 1rem;

            margin-top: auto;

        }

        @keyframes fadeIn {

            from {

                opacity: 0;

                transform: translateY(-10px);

            }

            to {

                opacity: 1;

                transform: translateY(0);

            }

        }

        /\* Responsive adjustments \*/

        @media (max-width: 768px) {

            .form-row,

            .form-buttons {

                flex-direction: column;

                gap: 0.5rem;

            }

            .container {

                padding: 1rem;

            }

            .content {

                padding: 1.5rem;

            }

            #search-input {

                width: 100%;

            }

            .action-buttons {

                flex-direction: column;

            }

        }

        /\* Theme toggle button styles \*/

        .theme-toggle {

            position: fixed;

            top: 1rem;

            right: 1rem;

            z-index: 1000;

            padding: 0.5rem 1rem;

            border-radius: 20px;

            border: 2px solid var(--primary);

            background: var(--card-bg);

            color: var(--primary);

            cursor: pointer;

            display: flex;

            align-items: center;

            gap: 0.5rem;

            transition: all 0.3s ease;

        }

        .theme-toggle:hover {

            background: var(--primary);

            color: var(--text-light);

        }

    </style>

</head>

<body>

    <button class="theme-toggle" onclick="toggleTheme()">

        <i class="fas fa-moon"></i> Dark Mode

    </button>

    <header>

        <h1><i class="fas fa-file-medical"></i> Medical Records Maintenance</h1>

    </header>

    <nav>

        <a href="dashboard.html"><i class="fas fa-home"></i> Dashboard</a>

        <a href="patients.html"><i class="fas fa-user-injured"></i> Patients</a>

        <a href="doctors.html"><i class="fas fa-user-md"></i> Doctors</a>

        <a href="appointments.html"><i class="fas fa-calendar-check"></i> Appointments</a>

    </nav>

    <div class="container">

        <div class="content">

            <h2>Add / Update Medical Record</h2>

            <form id="record-form">

                <div class="form-group">

                    <label for="record-id"><i class="fas fa-id-card"></i> Record ID</label>

                    <input type="text" id="record-id" name="record-id" readonly>

                </div>

                <div class="form-group">

                    <label for="patient-id"><i class="fas fa-user"></i> Patient</label>

                    <select id="patient-id" name="patient-id" required>

                        <option value="">Select Patient</option>

                        <!-- Options will be populated dynamically -->

                    </select>

                </div>

                <div class="form-group">

                    <label for="doctor-id"><i class="fas fa-user-md"></i> Doctor</label>

                    <select id="doctor-id" name="doctor-id" required>

                        <option value="">Select Doctor</option>

                        <!-- Options will be populated dynamically -->

                    </select>

                </div>

                <div class="form-group">

                    <label for="diagnosis"><i class="fas fa-notes-medical"></i> Diagnosis</label>

                    <textarea id="diagnosis" name="diagnosis" rows="3" required></textarea>

                </div>

                <div class="form-group">

                    <label for="treatment"><i class="fas fa-syringe"></i> Treatment</label>

                    <textarea id="treatment" name="treatment" rows="3"></textarea>

                </div>

                <div class="form-group">

                    <label for="record-date"><i class="fas fa-calendar"></i> Date</label>

                    <input type="date" id="record-date" name="record-date" required>

                </div>

                <div>

                    <button type="submit" id="save-record"><i class="fas fa-save"></i> Save</button>

                    <button type="button" id="clear-form" class="secondary"><i class="fas fa-eraser"></i> Clear</button>

                </div>

                <div class="message success-message" id="successMessage"><i class="fas fa-check-circle"></i> Record saved successfully!

                </div>

                <div class="message error-message" id="errorMessage"><i class="fas fa-exclamation-circle"></i> There was an error saving the record. Please try again.</div>

            </form>

        </div>

        <div class="content">

            <h2>Medical Records List</h2>

            <input type="text" id="search-input" placeholder="Search records by patient or doctor name">

            <table id="medical-records-table">

                <thead>

                    <tr>

                        <th>Record ID</th>

                        <th>Patient Name</th>

                        <th>Doctor Name</th>

                        <th>Date</th>

                        <th>Diagnosis</th>

                        <th>Actions</th>

                    </tr>

                </thead>

                <tbody>

                    <!-- Dynamic data will be populated here -->

                </tbody>

            </table>

        </div>

    </div>

    <footer>

        <p>&copy; 2025 Health Connect - Your Health, Our Priority</p>

    </footer>

    <script>

        function toggleTheme() {

            const html = document.documentElement;

            const currentTheme = html.getAttribute('data-theme');

            const newTheme = currentTheme === 'dark' ? 'light' : 'dark';

            html.setAttribute('data-theme', newTheme);

            localStorage.setItem('theme', newTheme);

            const themeToggle = document.querySelector('.theme-toggle');

            const icon = themeToggle.querySelector('i');

            icon.className = `fas ${newTheme === 'dark' ? 'fa-sun' : 'fa-moon'}`;

            themeToggle.innerHTML = `${icon.outerHTML} ${newTheme === 'dark' ? 'Light Mode' : 'Dark Mode'}`;

        }

        // Add this to your existing DOMContentLoaded event listener

        document.addEventListener('DOMContentLoaded', function() {

            // Set initial theme from localStorage

            const savedTheme = localStorage.getItem('theme') || 'light';

            document.documentElement.setAttribute('data-theme', savedTheme);

            // Update initial button state

            const themeToggle = document.querySelector('.theme-toggle');

            const icon = themeToggle.querySelector('i');

            icon.className = `fas ${savedTheme === 'dark' ? 'fa-sun' : 'fa-moon'}`;

            themeToggle.innerHTML = `${icon.outerHTML} ${savedTheme === 'dark' ? 'Light Mode' : 'Dark Mode'}`;

            const recordForm = document.getElementById('record-form');

            const patientDropdown = document.getElementById('patient-id');

            const doctorDropdown = document.getElementById('doctor-id');

            const recordsTableBody = document.querySelector('#medical-records-table tbody');

            const successMessage = document.getElementById('successMessage');

            const errorMessage = document.getElementById('errorMessage');

            const clearFormButton = document.getElementById('clear-form');

            // Fetch and populate patients

            function fetchPatients() {

                fetch('/api/patients')

                    .then(response => response.json())

                    .then(data => {

                        patientDropdown.innerHTML = '<option value="">Select Patient</option>';

                        data.forEach(patient => {

                            const option = document.createElement('option');

                            option.value = patient.id;

                            option.textContent = `${patient.first\_name} ${patient.last\_name}`;

                            patientDropdown.appendChild(option);

                        });

                    })

                    .catch(error => console.error('Error fetching patients:', error));

            }

            // Fetch and populate doctors

            function fetchDoctorsForDropdown() {

                const doctorDropdown = document.getElementById('doctor-id');

                doctorDropdown.innerHTML = '<option value="">Select Doctor</option>';

                fetch('/api/doctors/active')

                    .then(response => {

                        if (!response.ok) throw new Error('Network response was not ok');

                        return response.json();

                    })

                    .then(doctors => {

                        console.log('Received doctors:', doctors);

                        if (!doctors || doctors.length === 0) {

                            doctorDropdown.innerHTML += '<option disabled>No doctors available</option>';

                            return;

                        }

                        doctors.forEach(doctor => {

                            const option = document.createElement('option');

                            option.value = doctor.id;

                            option.textContent = `Dr. ${doctor.first\_name} ${doctor.last\_name}`;

                            doctorDropdown.appendChild(option);

                        });

                    })

                    .catch(error => {

                        console.error('Error fetching doctors:', error);

                        doctorDropdown.innerHTML = '<option value="">Error loading doctors</option>';

                    });

            }

            // Fetch and populate medical records

            function fetchMedicalRecords() {

                fetch('/api/medical-records')

                    .then(response => response.json())

                    .then(data => {

                        recordsTableBody.innerHTML = '';

                        data.forEach(record => {

                            const row = document.createElement('tr');

                            row.innerHTML = `

                                <td>${record.id}</td>

                                <td>${record.patient\_name}</td>

                                <td>${record.doctor\_name}</td>

                                <td>${record.date}</td>

                                <td>${record.diagnosis}</td>

                                <td class="action-buttons">

                                    <button class="edit-button" data-id="${record.id}"><i class="fas fa-edit"></i></button>

                                    <button class="delete-button delete" data-id="${record.id}"><i class="fas fa-trash"></i></button>

                                </td>

                            `;

                            recordsTableBody.appendChild(row);

                        });

                        attachEditAndDeleteListeners();

                    })

                    .catch(error => console.error('Error fetching medical records:', error));

            }

            // Attach listeners to edit and delete buttons

            function attachEditAndDeleteListeners() {

                const editButtons = document.querySelectorAll('.edit-button');

                const deleteButtons = document.querySelectorAll('.delete-button');

                editButtons.forEach(button => {

                    button.addEventListener('click', function() {

                        const id = this.getAttribute('data-id');

                        fetch(`/api/medical-records/${id}`)

                            .then(response => response.json())

                            .then(record => {

                                document.getElementById('record-id').value = record.id;

                                document.getElementById('patient-id').value = record.patient\_id;

                                document.getElementById('doctor-id').value = record.doctor\_id;

                                document.getElementById('diagnosis').value = record.diagnosis;

                                document.getElementById('treatment').value = record.treatment;

                                document.getElementById('record-date').value = record.date;

                            })

                            .catch(error => console.error('Error fetching medical record details:', error));

                    });

                });

                deleteButtons.forEach(button => {

                    button.addEventListener('click', function() {

                        const id = this.getAttribute('data-id');

                        if (confirm('Are you sure you want to delete this medical record?')) {

                            fetch(`/api/medical-records/${id}`, {

                                    method: 'DELETE'

                                })

                                .then(response => response.json())

                                .then(data => {

                                    if (data.success) {

                                        fetchMedicalRecords();

                                        alert('Medical record deleted successfully');

                                    }

                                })

                                .catch(error => console.error('Error deleting medical record:', error));

                        }

                    });

                });

            }

            // Save medical record

            recordForm.addEventListener('submit', function(event) {

                event.preventDefault();

                const formData = new FormData(recordForm);

                const recordData = {

                    id: formData.get('record-id') || null,

                    patient\_id: formData.get('patient-id'),

                    doctor\_id: formData.get('doctor-id'),

                    diagnosis: formData.get('diagnosis'),

                    treatment: formData.get('treatment'),

                    date: formData.get('record-date')

                };

                fetch('/api/medical-records', {

                        method: 'POST',

                        headers: {

                            'Content-Type': 'application/json'

                        },

                        body: JSON.stringify(recordData)

                    })

                    .then(response => response.json())

                    .then(data => {

                        if (data.success) {

                            successMessage.style.display = 'block';

                            errorMessage.style.display = 'none';

                            recordForm.reset();

                            fetchMedicalRecords();

                        } else {

                            successMessage.style.display = 'none';

                            errorMessage.style.display = 'block';

                        }

                    })

                    .catch(error => {

                        successMessage.style.display = 'none';

                        errorMessage.style.display = 'block';

                        console.error('Error saving medical record:', error);

                    });

            });

            // Clear form

            clearFormButton.addEventListener('click', function() {

                recordForm.reset();

                document.getElementById('record-id').value = '';

                successMessage.style.display = 'none';

                errorMessage.style.display = 'none';

            });

            // Initial fetches

            fetchPatients();

            fetchDoctorsForDropdown();

            fetchMedicalRecords();

        });

    </script>

</body>

</html>

**Billing.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Billing Management - Health Connect</title>

    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

    <style>

         :root {

            --primary: #2c9caf;

            --primary-dark: #1f7a89;

            --secondary: #ff6b6b;

            --text-dark: #333;

            --light-bg: #f9fafb;

            --card-bg: #ffffff;

            --success: #4CAF50;

            --error: #f44336;

            --warning: #ff9800;

            --info: #2196F3;

        }

        [data-theme="dark"] {

            --primary: #3fb8cc;

            --primary-dark: #2c9caf;

            --text-dark: #e0e0e0;

            --text-light: #ffffff;

            --light-bg: #121212;

            --card-bg: #1e1e1e;

            --border-color: #404040;

            --hover-bg: rgba(44, 156, 175, 0.1);

            --table-header: #2d2d2d;

        }

        [data-theme="dark"] body {

            background-color: var(--light-bg);

            color: var(--text-light);

        }

        [data-theme="dark"] .card {

            background-color: var(--card-bg);

            box-shadow: 0 5px 15px rgba(0, 0, 0, 0.3);

        }

        [data-theme="dark"] .summary-card {

            background-color: var(--card-bg);

        }

        [data-theme="dark"] input,

        [data-theme="dark"] select,

        [data-theme="dark"] textarea {

            background-color: var(--card-bg);

            color: var(--text-light);

            border-color: var(--border-color);

        }

        [data-theme="dark"] label {

            color: var(--text-light);

        }

        [data-theme="dark"] table {

            background-color: var(--card-bg);

            border-color: var(--border-color);

        }

        [data-theme="dark"] th {

            background-color: var(--table-header);

            color: var(--text-light);

        }

        [data-theme="dark"] td {

            border-color: var(--border-color);

            color: var(--text-light);

        }

        [data-theme="dark"] tr:hover {

            background-color: var(--hover-bg);

        }

        [data-theme="dark"] .summary-details h3 {

            color: #aaa;

        }

        [data-theme="dark"] .summary-details p {

            color: var(--text-light);

        }

        body {

            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

            background-color: var(--light-bg);

            margin: 0;

            padding: 0;

            display: flex;

            flex-direction: column;

            min-height: 100vh;

            color: var(--text-dark);

        }

        header {

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            padding: 1.5rem;

            text-align: center;

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

        }

        header h1 {

            margin: 0;

            font-size: 2.2rem;

            font-weight: 600;

            letter-spacing: 0.5px;

        }

        nav {

            background-color: #2d3748;

            display: flex;

            justify-content: center;

            padding: 0.7rem;

            box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

        }

        nav a {

            color: white;

            text-decoration: none;

            padding: 0.7rem 1.5rem;

            margin: 0 0.5rem;

            border-radius: 4px;

            transition: all 0.3s ease;

            font-weight: 500;

        }

        nav a:hover {

            background-color: rgba(255, 255, 255, 0.1);

            transform: translateY(-2px);

        }

        main {

            flex: 1;

            padding: 2rem;

        }

        .card {

            background-color: var(--card-bg);

            border-radius: 12px;

            box-shadow: 0 5px 15px rgba(0, 0, 0, 0.08);

            margin-bottom: 2rem;

            transition: transform 0.3s ease;

            overflow: hidden;

        }

        .card:hover {

            transform: translateY(-5px);

        }

        .card-header {

            background: linear-gradient(to right, var(--primary), var(--primary-dark));

            color: white;

            padding: 1.2rem 1.5rem;

            font-size: 1.2rem;

            font-weight: 600;

            display: flex;

            align-items: center;

        }

        .card-header i {

            margin-right: 0.7rem;

            font-size: 1.1em;

        }

        .card-body {

            padding: 1.5rem;

        }

        .form-group {

            margin-bottom: 1.2rem;

        }

        label {

            display: block;

            margin-bottom: 0.7rem;

            font-weight: 600;

            color: var(--text-dark);

        }

        input,

        select,

        textarea {

            width: 100%;

            padding: 0.75rem;

            border: 1px solid #ddd;

            border-radius: 6px;

            font-size: 1rem;

            transition: border 0.3s ease, box-shadow 0.3s ease;

        }

        input:focus,

        select:focus,

        textarea:focus {

            outline: none;

            border-color: var(--primary);

            box-shadow: 0 0 0 3px rgba(44, 156, 175, 0.2);

        }

        .form-row {

            display: flex;

            gap: 1rem;

        }

        .form-row .form-group {

            flex: 1;

        }

        .form-actions {

            display: flex;

            justify-content: space-between;

            margin-top: 1.5rem;

        }

        button {

            padding: 0.8rem 1.5rem;

            border: none;

            border-radius: 6px;

            cursor: pointer;

            font-size: 1rem;

            font-weight: 600;

            transition: all 0.3s ease;

            display: inline-flex;

            align-items: center;

            justify-content: center;

        }

        button i {

            margin-right: 0.5rem;

        }

        .btn-primary {

            background-color: var(--primary);

            color: white;

            box-shadow: 0 4px 6px rgba(44, 156, 175, 0.2);

        }

        .btn-primary:hover {

            background-color: var(--primary-dark);

            transform: translateY(-2px);

            box-shadow: 0 6px 8px rgba(44, 156, 175, 0.3);

        }

        .btn-danger {

            background-color: var(--error);

            color: white;

            box-shadow: 0 4px 6px rgba(244, 67, 54, 0.2);

        }

        .btn-danger:hover {

            background-color: #d32f2f;

            transform: translateY(-2px);

            box-shadow: 0 6px 8px rgba(244, 67, 54, 0.3);

        }

        .search-container {

            position: relative;

            margin-bottom: 1.5rem;

        }

        .search-container i {

            position: absolute;

            left: 1rem;

            top: 50%;

            transform: translateY(-50%);

            color: #aaa;

        }

        #search-input {

            padding-left: 2.5rem;

            width: 100%;

            box-sizing: border-box;

        }

        table {

            width: 100%;

            border-collapse: collapse;

            margin-top: 1rem;

            background-color: white;

            border-radius: 6px;

            overflow: hidden;

        }

        thead {

            background-color: var(--primary);

            color: white;

        }

        th,

        td {

            padding: 1rem;

            text-align: left;

            border-bottom: 1px solid #eee;

        }

        tbody tr:hover {

            background-color: rgba(44, 156, 175, 0.05);

        }

        .status-pill {

            display: inline-block;

            padding: 0.3rem 1rem;

            border-radius: 50px;

            font-size: 0.85rem;

            font-weight: 600;

        }

        .status-paid {

            background-color: rgba(76, 175, 80, 0.1);

            color: var(--success);

        }

        .status-unpaid {

            background-color: rgba(244, 67, 54, 0.1);

            color: var(--error);

        }

        .status-pending {

            background-color: rgba(255, 152, 0, 0.1);

            color: var(--warning);

        }

        .action-buttons {

            display: flex;

            gap: 0.5rem;

        }

        .btn-action {

            padding: 0.4rem 0.7rem;

            font-size: 0.85rem;

            border-radius: 4px;

        }

        .btn-edit {

            background-color: var(--info);

            color: white;

        }

        .btn-delete {

            background-color: var(--error);

            color: white;

        }

        .btn-view {

            background-color: var(--primary);

            color: white;

        }

        .btn-print {

            background-color: var(--warning);

            color: white;

        }

        .summary-cards {

            display: flex;

            gap: 1rem;

            margin-bottom: 2rem;

        }

        .summary-card {

            flex: 1;

            background-color: white;

            border-radius: 10px;

            padding: 1.5rem;

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.07);

            display: flex;

            align-items: center;

        }

        .summary-icon {

            width: 50px;

            height: 50px;

            border-radius: 10px;

            display: flex;

            align-items: center;

            justify-content: center;

            margin-right: 1rem;

            font-size: 1.5rem;

        }

        .summary-icon.total {

            background-color: rgba(33, 150, 243, 0.1);

            color: var(--info);

        }

        .summary-icon.paid {

            background-color: rgba(76, 175, 80, 0.1);

            color: var(--success);

        }

        .summary-icon.pending {

            background-color: rgba(255, 152, 0, 0.1);

            color: var(--warning);

        }

        .summary-icon.unpaid {

            background-color: rgba(244, 67, 54, 0.1);

            color: var(--error);

        }

        .summary-details h3 {

            margin: 0;

            color: #888;

            font-size: 0.9rem;

            font-weight: 500;

        }

        .summary-details p {

            margin: 0.3rem 0 0 0;

            font-size: 1.5rem;

            font-weight: 600;

            color: var(--text-dark);

        }

        .pagination {

            display: flex;

            justify-content: center;

            margin-top: 2rem;

        }

        .pagination button {

            background-color: white;

            color: var(--text-dark);

            border: 1px solid #ddd;

            padding: 0.5rem 1rem;

            margin: 0 0.2rem;

            cursor: pointer;

            transition: all 0.3s ease;

        }

        .pagination button.active {

            background-color: var(--primary);

            color: white;

            border-color: var(--primary);

        }

        .pagination button:hover:not(.active) {

            background-color: #f5f5f5;

        }

        footer {

            background-color: #2d3748;

            color: white;

            text-align: center;

            padding: 1rem;

            margin-top: auto;

        }

        /\* Responsive adjustments \*/

        @media (max-width: 992px) {

            .summary-cards {

                flex-wrap: wrap;

            }

            .summary-card {

                flex: 1 1 calc(50% - 1rem);

            }

        }

        @media (max-width: 768px) {

            .form-row {

                flex-direction: column;

                gap: 0;

            }

            .summary-card {

                flex: 1 1 100%;

            }

            main {

                padding: 1rem;

            }

            th,

            td {

                padding: 0.75rem 0.5rem;

            }

            .action-buttons {

                flex-direction: column;

                gap: 0.3rem;

            }

        }

        /\* Theme toggle button styles \*/

        .theme-toggle {

            position: fixed;

            top: 1rem;

            right: 1rem;

            z-index: 1000;

            padding: 0.5rem 1rem;

            border-radius: 20px;

            border: 2px solid var(--primary);

            background: var(--card-bg);

            color: var(--primary);

            cursor: pointer;

            display: flex;

            align-items: center;

            gap: 0.5rem;

            transition: all 0.3s ease;

        }

        .theme-toggle:hover {

            background: var(--primary);

            color: var(--text-light);

        }

    </style>

</head>

<body>

    <button class="theme-toggle" onclick="toggleTheme()">

        <i class="fas fa-moon"></i> Dark Mode

    </button>

    <header>

        <h1><i class="fas fa-file-invoice-dollar"></i> Billing Management</h1>

    </header>

    <nav>

        <a href="dashboard.html"><i class="fas fa-home"></i> Dashboard</a>

        <a href="appointments.html"><i class="fas fa-calendar-check"></i> Appointments</a>

        <a href="patients.html"><i class="fas fa-user-injured"></i> Patients</a>

        <a href="doctors.html"><i class="fas fa-user-md"></i> Doctors</a>

        <a href="medical\_records.html"><i class="fas fa-notes-medical"></i> Records</a>

    </nav>

    <main>

        <div class="summary-cards">

            <div class="summary-card">

                <div class="summary-icon total">

                    <i class="fas fa-chart-pie"></i>

                </div>

                <div class="summary-details">

                    <h3>Total Billing</h3>

                    <p id="total-billing">₹0.00</p>

                </div>

            </div>

            <div class="summary-card">

                <div class="summary-icon paid">

                    <i class="fas fa-check-circle"></i>

                </div>

                <div class="summary-details">

                    <h3>Paid Amount</h3>

                    <p id="paid-amount">₹0.00</p>

                </div>

            </div>

            <div class="summary-card">

                <div class="summary-icon pending">

                    <i class="fas fa-clock"></i>

                </div>

                <div class="summary-details">

                    <h3>Pending Amount</h3>

                    <p id="pending-amount">₹0.00</p>

                </div>

            </div>

            <div class="summary-card">

                <div class="summary-icon unpaid">

                    <i class="fas fa-exclamation-circle"></i>

                </div>

                <div class="summary-details">

                    <h3>Unpaid Amount</h3>

                    <p id="unpaid-amount">₹0.00</p>

                </div>

            </div>

        </div>

        <div class="card">

            <div class="card-header">

                <i class="fas fa-plus-circle"></i> Add/Update Billing Record

            </div>

            <div class="card-body">

                <form id="billing-form">

                    <div class="form-row">

                        <div class="form-group">

                            <label for="billing-id">Billing ID</label>

                            <input type="text" id="billing-id" name="billing-id" readonly>

                        </div>

                        <div class="form-group">

                            <label for="appointment-id">Appointment ID</label>

                            <input type="text" id="appointment-id" name="appointment-id" required>

                        </div>

                    </div>

                    <div class="form-row">

                        <div class="form-group">

                            <label for="patient-name">Patient Name</label>

                            <input type="text" id="patient-name" name="patient-name" required>

                        </div>

                        <div class="form-group">

                            <label for="doctor-name">Doctor Name</label>

                            <input type="text" id="doctor-name" name="doctor-name" required>

                        </div>

                    </div>

                    <div class="form-row">

                        <div class="form-group">

                            <label for="appointment-date">Appointment Date</label>

                            <input type="date" id="appointment-date" name="appointment-date" required>

                        </div>

                        <div class="form-group">

                            <label for="billing-date">Billing Date</label>

                            <input type="date" id="billing-date" name="billing-date" required>

                        </div>

                    </div>

                    <div class="form-row">

                        <div class="form-group">

                            <label for="service-details">Service Details</label>

                            <textarea id="service-details" name="service-details" rows="3" placeholder="Enter service details"></textarea>

                        </div>

                    </div>

                    <div class="form-row">

                        <div class="form-group">

                            <label for="billing-amount">Billing Amount (₹)</label>

                            <input type="number" id="billing-amount" name="billing-amount" min="0" step="0.01" required>

                        </div>

                        <div class="form-group">

                            <label for="discount">Discount (₹)</label>

                            <input type="number" id="discount" name="discount" min="0" step="0.01" value="0">

                        </div>

                        <div class="form-group">

                            <label for="tax">Tax (%)</label>

                            <input type="number" id="tax" name="tax" min="0" max="100" step="0.01" value="18">

                        </div>

                    </div>

                    <div class="form-group">

                        <label for="total-amount">Total Amount (₹)</label>

                        <input type="number" id="total-amount" name="total-amount" readonly>

                    </div>

                    <div class="form-group">

                        <label for="billing-status">Billing Status</label>

                        <select id="billing-status" name="billing-status" required>

                            <option value="Paid">Paid</option>

                            <option value="Unpaid">Unpaid</option>

                            <option value="Pending">Pending</option>

                        </select>

                    </div>

                    <div class="form-group">

                        <label for="payment-method">Payment Method</label>

                        <select id="payment-method" name="payment-method">

                            <option value="">Select Payment Method</option>

                            <option value="Cash">Cash</option>

                            <option value="Credit Card">Credit Card</option>

                            <option value="Debit Card">Debit Card</option>

                            <option value="UPI">UPI</option>

                            <option value="Insurance">Insurance</option>

                            <option value="Bank Transfer">Bank Transfer</option>

                        </select>

                    </div>

                    <div class="form-actions">

                        <button type="submit" id="save-billing" class="btn-primary"><i class="fas fa-save"></i> Save</button>

                        <button type="button" id="clear-form" class="btn-danger"><i class="fas fa-times"></i> Clear</button>

                    </div>

                </form>

            </div>

        </div>

        <div class="card">

            <div class="card-header">

                <i class="fas fa-list"></i> Billing Records

            </div>

            <div class="card-body">

                <div class="search-container">

                    <i class="fas fa-search"></i>

                    <input type="text" id="search-input" placeholder="Search billing records by patient name, ID, status...">

                </div>

                <table id="billing-table">

                    <thead>

                        <tr>

                            <th>Bill ID</th>

                            <th>Patient</th>

                            <th>Doctor</th>

                            <th>Date</th>

                            <th>Amount (₹)</th>

                            <th>Status</th>

                            <th>Actions</th>

                        </tr>

                    </thead>

                    <tbody id="billing-records">

                        <!-- Records will be populated dynamically -->

                    </tbody>

                </table>

                <div class="pagination" id="pagination">

                    <!-- Pagination controls will be added dynamically -->

                </div>

            </div>

        </div>

    </main>

    <footer>

        <p>&copy; 2025 Health Connect - Your Health, Our Priority</p>

    </footer>

    <script>

        function toggleTheme() {

            const html = document.documentElement;

            const currentTheme = html.getAttribute('data-theme');

            const newTheme = currentTheme === 'dark' ? 'light' : 'dark';

            html.setAttribute('data-theme', newTheme);

            localStorage.setItem('theme', newTheme);

            const themeToggle = document.querySelector('.theme-toggle');

            const icon = themeToggle.querySelector('i');

            icon.className = `fas ${newTheme === 'dark' ? 'fa-sun' : 'fa-moon'}`;

            themeToggle.innerHTML = `${icon.outerHTML} ${newTheme === 'dark' ? 'Light Mode' : 'Dark Mode'}`;

        }

        // Add this to your existing DOMContentLoaded event listener

        document.addEventListener('DOMContentLoaded', function() {

            // Set initial theme from localStorage

            const savedTheme = localStorage.getItem('theme') || 'light';

            document.documentElement.setAttribute('data-theme', savedTheme);

            // Update initial button state

            const themeToggle = document.querySelector('.theme-toggle');

            const icon = themeToggle.querySelector('i');

            icon.className = `fas ${savedTheme === 'dark' ? 'fa-sun' : 'fa-moon'}`;

            themeToggle.innerHTML = `${icon.outerHTML} ${savedTheme === 'dark' ? 'Light Mode' : 'Dark Mode'}`;

            // Calculate total amount when input values change

            const calculateTotal = () => {

                const amount = parseFloat(document.getElementById('billing-amount').value) || 0;

                const discount = parseFloat(document.getElementById('discount').value) || 0;

                const taxPercent = parseFloat(document.getElementById('tax').value) || 0;

                const subtotal = amount - discount;

                const tax = subtotal \* (taxPercent / 100);

                const total = subtotal + tax;

                document.getElementById('total-amount').value = total.toFixed(2);

            };

            document.getElementById('billing-amount').addEventListener('input', calculateTotal);

            document.getElementById('discount').addEventListener('input', calculateTotal);

            document.getElementById('tax').addEventListener('input', calculateTotal);

            // Set current date for billing date

            const today = new Date();

            const yyyy = today.getFullYear();

            const mm = String(today.getMonth() + 1).padStart(2, '0');

            const dd = String(today.getDate()).padStart(2, '0');

            document.getElementById('billing-date').value = `${yyyy}-${mm}-${dd}`;

            // Clear form

            document.getElementById('clear-form').addEventListener('click', function() {

                document.getElementById('billing-form').reset();

                document.getElementById('billing-id').value = '';

                document.getElementById('billing-date').value = `${yyyy}-${mm}-${dd}`;

            });

            // Submit form

            document.getElementById('billing-form').addEventListener('submit', function(e) {

                e.preventDefault();

                const formData = new FormData(this);

                const billingData = {

                    id: formData.get('billing-id'),

                    appointment\_id: formData.get('appointment-id'),

                    patient\_name: formData.get('patient-name'),

                    doctor\_name: formData.get('doctor-name'),

                    appointment\_date: formData.get('appointment-date'),

                    billing\_date: formData.get('billing-date'),

                    service\_details: formData.get('service-details'),

                    amount: formData.get('billing-amount'),

                    discount: formData.get('discount'),

                    tax: formData.get('tax'),

                    total\_amount: formData.get('total-amount'),

                    status: formData.get('billing-status'),

                    payment\_method: formData.get('payment-method')

                };

                const url = billingData.id ? `/api/billing/${billingData.id}` : '/api/billing';

                const method = billingData.id ? 'PUT' : 'POST';

                fetch(url, {

                        method: method,

                        headers: {

                            'Content-Type': 'application/json'

                        },

                        body: JSON.stringify(billingData)

                    })

                    .then(response => response.json())

                    .then(data => {

                        if (data.success) {

                            alert('Billing record saved successfully!');

                            document.getElementById('clear-form').click();

                            loadBillingRecords();

                            updateSummary();

                        } else {

                            alert('Error: ' + data.message);

                        }

                    })

                    .catch(error => {

                        console.error('Error:', error);

                        alert('An error occurred while saving the billing record.');

                    });

            });

            // Search functionality

            document.getElementById('search-input').addEventListener('input', function() {

                const searchTerm = this.value.toLowerCase();

                const rows = document.querySelectorAll('#billing-records tr');

                rows.forEach(row => {

                    const text = row.textContent.toLowerCase();

                    row.style.display = text.includes(searchTerm) ? '' : 'none';

                });

            });

            // Load billing records

            function loadBillingRecords() {

                fetch('/api/billing')

                    .then(response => response.json())

                    .then(data => {

                        const tbody = document.getElementById('billing-records');

                        tbody.innerHTML = '';

                        if (data.length === 0) {

                            const row = document.createElement('tr');

                            row.innerHTML = '<td colspan="7" style="text-align: center;">No billing records found</td>';

                            tbody.appendChild(row);

                            return;

                        }

                        data.forEach(record => {

                            const row = document.createElement('tr');

                            // Determine status class

                            let statusClass = '';

                            switch (record.status) {

                                case 'Paid':

                                    statusClass = 'status-paid';

                                    break;

                                case 'Unpaid':

                                    statusClass = 'status-unpaid';

                                    break;

                                case 'Pending':

                                    statusClass = 'status-pending';

                                    break;

                            }

                            row.innerHTML = `

                                <td>${record.id}</td>

                                <td>${record.patient\_name}</td>

                                <td>${record.doctor\_name}</td>

                                <td>${record.billing\_date}</td>

                                <td>₹${parseFloat(record.total\_amount).toFixed(2)}</td>

                                <td><span class="status-pill ${statusClass}">${record.status}</span></td>

                                <td>

                                    <div class="action-buttons">

                                        <button class="btn-action btn-view" data-id="${record.id}"><i class="fas fa-eye"></i></button>

                                        <button class="btn-action btn-edit" data-id="${record.id}"><i class="fas fa-edit"></i></button>

                                        <button class="btn-action btn-print" data-id="${record.id}"><i class="fas fa-print"></i></button>

                                        <button class="btn-action btn-delete" data-id="${record.id}"><i class="fas fa-trash"></i></button>

                                    </div>

                                </td>

                            `;

                            tbody.appendChild(row);

                        });

                        // Add event listeners to action buttons

                        document.querySelectorAll('.btn-edit').forEach(button => {

                            button.addEventListener('click', function() {

                                const id = this.getAttribute('data-id');

                                editBillingRecord(id);

                            });

                        });

                        document.querySelectorAll('.btn-delete').forEach(button => {

                            button.addEventListener('click', function() {

                                const id = this.getAttribute('data-id');

                                if (confirm('Are you sure you want to delete this billing record?')) {

                                    deleteBillingRecord(id);

                                }

                            });

                        });

                        document.querySelectorAll('.btn-print').forEach(button => {

                            button.addEventListener('click', function() {

                                const id = this.getAttribute('data-id');

                                window.open(`/api/billing/${id}/invoice`, '\_blank');

                            });

                        });

                        document.querySelectorAll('.btn-view').forEach(button => {

                            button.addEventListener('click', function() {

                                const id = this.getAttribute('data-id');

                                viewBillingRecord(id);

                            });

                        });

                    })

                    .catch(error => {

                        console.error('Error loading billing records:', error);

                    });

            }

            // Edit billing record

            function editBillingRecord(id) {

                fetch(`/api/billing/${id}`)

                    .then(response => response.json())

                    .then(data => {

                        document.getElementById('billing-id').value = data.id;

                        document.getElementById('appointment-id').value = data.appointment\_id;

                        document.getElementById('patient-name').value = data.patient\_name;

                        document.getElementById('doctor-name').value = data.doctor\_name;

                        document.getElementById('appointment-date').value = data.appointment\_date;

                        document.getElementById('billing-date').value = data.billing\_date;

                        document.getElementById('service-details').value = data.service\_details;

                        document.getElementById('billing-amount').value = data.amount;

                        document.getElementById('discount').value = data.discount;

                        document.getElementById('tax').value = data.tax;

                        document.getElementById('total-amount').value = data.total\_amount;

                        document.getElementById('billing-status').value = data.status;

                        document.getElementById('payment-method').value = data.payment\_method;

                        // Scroll to form

                        document.querySelector('.card').scrollIntoView({

                            behavior: 'smooth'

                        });

                    })

                    .catch(error => {

                        console.error('Error fetching billing record:', error);

                    });

            }

            // Delete billing record

            function deleteBillingRecord(id) {

                fetch(`/api/billing/${id}`, {

                        method: 'DELETE'

                    })

                    .then(response => response.json())

                    .then(data => {

                        if (data.success) {

                            alert('Billing record deleted successfully!');

                            loadBillingRecords();

                            updateSummary();

                        } else {

                            alert('Error: ' + data.message);

                        }

                    })

                    .catch(error => {

                        console.error('Error:', error);

                        alert('An error occurred while deleting the billing record.');

                    });

            }

            // View billing record details

            function viewBillingRecord(id) {

                window.location.href = `billing\_details.html?id=${id}`;

            }

            // Update summary cards

            function updateSummary() {

                fetch('/api/billing/summary')

                    .then(response => response.json())

                    .then(data => {

                        document.getElementById('total-billing').textContent = `₹${data.total.toFixed(2)}`;

                        document.getElementById('paid-amount').textContent = `₹${data.paid.toFixed(2)}`;

                        document.getElementById('pending-amount').textContent = `₹${data.pending.toFixed(2)}`;

                        document.getElementById('unpaid-amount').textContent = `₹${data.unpaid.toFixed(2)}`;

                    })

                    .catch(error => {

                        console.error('Error fetching billing summary:', error);

                    });

            }

            // Initial load

            loadBillingRecords();

            updateSummary();

            // Auto-fill patient and doctor when appointment ID is entered

            document.getElementById('appointment-id').addEventListener('blur', function() {

                const appointmentId = this.value;

                if (appointmentId) {

                    fetch(`/api/appointments/${appointmentId}`)

                        .then(response => response.json())

                        .then(data => {

                            if (data) {

                                document.getElementById('patient-name').value = data.patient\_name;

                                document.getElementById('doctor-name').value = data.doctor\_name;

                                document.getElementById('appointment-date').value = data.appointment\_date;

                            }

                        })

                        .catch(error => {

                            console.error('Error fetching appointment:', error);

                        });

                }

            });

        });

    </script>

</body>

</html>

**Notification.html**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Notifications - Health Connect</title>

    <link href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0/css/all.min.css" rel="stylesheet">

    <link href="css/themes.css" rel="stylesheet">

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

    <style>

         :root {

            --primary: #2c9caf;

            --primary-dark: #1f7a89;

            --secondary: #ff6b6b;

            --text-dark: #333;

            --text-light: #ffffff;

            --light-bg: #f9fafb;

            --card-bg: #ffffff;

            --success: #4CAF50;

            --error: #f44336;

            --border-color: #ddd;

            --table-header: #f5f5f5;

            --hover-bg: rgba(44, 156, 175, 0.05);

        }

        body {

            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

            background-color: var(--light-bg);

            margin: 0;

            padding: 0;

            display: flex;

            flex-direction: column;

            min-height: 100vh;

            color: var(--text-dark);

            transition: background-color 0.3s ease, color 0.3s ease;

        }

        header {

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            padding: 1.5rem;

            text-align: center;

            position: relative;

            box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);

        }

        header h1 {

            margin: 0;

            font-size: 2.2rem;

            font-weight: 600;

            letter-spacing: 0.5px;

        }

        nav {

            background-color: #2d3748;

            display: flex;

            justify-content: center;

            padding: 0.7rem;

            box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

        }

        nav a {

            color: white;

            text-decoration: none;

            padding: 0.7rem 1.5rem;

            margin: 0 0.5rem;

            border-radius: 4px;

            transition: all 0.3s ease;

            font-weight: 500;

        }

        nav a:hover {

            background-color: rgba(255, 255, 255, 0.1);

            transform: translateY(-2px);

        }

        .container {

            flex: 1;

            padding: 2.5rem;

            display: flex;

            flex-direction: column;

            align-items: center;

        }

        .content {

            width: 100%;

            max-width: 1000px;

            background-color: var(--card-bg);

            padding: 2.5rem;

            border-radius: 12px;

            box-shadow: 0 5px 15px rgba(0, 0, 0, 0.08);

            margin-bottom: 2rem;

            transition: background-color 0.3s ease, border-color 0.3s ease, color 0.3s ease;

        }

        .content h2 {

            color: var(--primary);

            margin-top: 0;

            font-size: 1.8rem;

            position: relative;

            padding-bottom: 0.5rem;

            margin-bottom: 1.5rem;

        }

        .content h2::after {

            content: '';

            position: absolute;

            left: 0;

            bottom: 0;

            height: 3px;

            width: 60px;

            background-color: var(--secondary);

        }

        table {

            width: 100%;

            border-collapse: collapse;

            margin-top: 1.5rem;

            box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);

        }

        th,

        td {

            padding: 1rem;

            border: 1px solid #ddd;

            text-align: left;

        }

        th {

            background-color: var(--light-bg);

            font-weight: 600;

        }

        tr:hover {

            background-color: rgba(44, 156, 175, 0.05);

        }

        .action-buttons {

            display: flex;

            gap: 0.5rem;

        }

        .action-buttons button {

            padding: 0.5rem;

            margin: 0;

        }

        button {

            background: linear-gradient(135deg, var(--primary) 0%, var(--primary-dark) 100%);

            color: white;

            border: none;

            padding: 0.8rem 1.5rem;

            border-radius: 6px;

            cursor: pointer;

            font-size: 1rem;

            font-weight: 600;

            transition: all 0.3s ease;

            display: inline-flex;

            align-items: center;

            justify-content: center;

            box-shadow: 0 4px 6px rgba(44, 156, 175, 0.2);

            margin-right: 0.5rem;

        }

        button i {

            margin-right: 0.5rem;

        }

        button:hover {

            background: linear-gradient(135deg, var(--primary-dark) 0%, var(--primary) 100%);

            transform: translateY(-2px);

            box-shadow: 0 6px 8px rgba(44, 156, 175, 0.3);

        }

        button.secondary {

            background: #e0e0e0;

            color: #333;

        }

        button.secondary:hover {

            background: #d0d0d0;

        }

        .filters {

            display: flex;

            flex-wrap: wrap;

            gap: 1rem;

            margin-bottom: 1.5rem;

        }

        .filters select {

            padding: 0.75rem;

            border: 1px solid #ddd;

            border-radius: 6px;

            font-size: 1rem;

            flex-grow: 1;

        }

        #search-input {

            margin: 1.5rem 0;

            padding: 0.75rem;

            border: 1px solid #ddd;

            border-radius: 6px;

            width: 50%;

        }

        .badge {

            display: inline-block;

            padding: 0.25rem 0.75rem;

            border-radius: 999px;

            font-size: 0.75rem;

            font-weight: 600;

            text-transform: uppercase;

        }

        .badge-upcoming {

            background-color: #e3f2fd;

            color: #1565c0;

        }

        .badge-today {

            background-color: #fff8e1;

            color: #f57f17;

        }

        .badge-urgent {

            background-color: #ffebee;

            color: #c62828;

        }

        footer {

            background-color: #2d3748;

            color: white;

            text-align: center;

            padding: 1rem;

            margin-top: auto;

        }

        @keyframes fadeIn {

            from {

                opacity: 0;

                transform: translateY(-10px);

            }

            to {

                opacity: 1;

                transform: translateY(0);

            }

        }

        .notification-info {

            display: flex;

            align-items: center;

        }

        .notification-info i {

            margin-right: 0.5rem;

            font-size: 1.2rem;

        }

        /\* Responsive adjustments \*/

        @media (max-width: 768px) {

            .container {

                padding: 1rem;

            }

            .content {

                padding: 1.5rem;

            }

            #search-input {

                width: 100%;

            }

            .action-buttons {

                flex-direction: column;

            }

            .filters {

                flex-direction: column;

            }

            table {

                font-size: 0.9rem;

            }

            th,

            td {

                padding: 0.75rem 0.5rem;

            }

        }

        /\* Add smooth transitions for theme changes \*/

        body,

        .content,

        table,

        th,

        td,

        button,

        input,

        select,

        nav,

        footer {

            transition: all 0.3s ease;

        }

        /\* Dark mode specific adjustments for notifications page \*/

        [data-theme="dark"] {

            --primary: #3fb8cc;

            --primary-dark: #2c9caf;

            --text-dark: #e0e0e0;

            --text-light: #ffffff;

            --light-bg: #121212;

            --card-bg: #1e1e1e;

            --border-color: #404040;

            --table-header: #2d2d2d;

            --hover-bg: rgba(44, 156, 175, 0.1);

        }

        [data-theme="dark"] body {

            background-color: var(--light-bg);

            color: var(--text-light);

        }

        [data-theme="dark"] .content {

            background-color: var(--card-bg);

            box-shadow: 0 5px 15px rgba(0, 0, 0, 0.3);

        }

        [data-theme="dark"] table {

            border-color: var(--border-color);

        }

        [data-theme="dark"] th {

            background-color: var(--table-header);

            color: var(--text-light);

            border-color: var(--border-color);

        }

        [data-theme="dark"] td {

            border-color: var(--border-color);

            color: var(--text-light);

        }

        [data-theme="dark"] tr:hover {

            background-color: var(--hover-bg);

        }

        [data-theme="dark"] input,

        [data-theme="dark"] select {

            background-color: var(--card-bg);

            color: var(--text-light);

            border-color: var(--border-color);

        }

        [data-theme="dark"] button.secondary {

            background: #404040;

            color: var(--text-light);

        }

        [data-theme="dark"] button.secondary:hover {

            background: #505050;

        }

        [data-theme="dark"] .badge {

            opacity: 0.9;

        }

        [data-theme="dark"] .badge-upcoming {

            background-color: #1565c0;

            color: #e3f2fd;

        }

        [data-theme="dark"] .badge-today {

            background-color: #f57f17;

            color: #fff8e1;

        }

        [data-theme="dark"] .badge-urgent {

            background-color: #c62828;

            color: #ffebee;

        }

        [data-theme="dark"] nav {

            background-color: var(--card-bg);

            border-bottom: 1px solid var(--border-color);

        }

        [data-theme="dark"] footer {

            background-color: var(--card-bg);

            border-top: 1px solid var(--border-color);

        }

    </style>

</head>

<body>

    <button class="theme-toggle" onclick="toggleTheme()" style="position: fixed; top: 20px; right: 20px; z-index: 1000;">

        <i class="fas fa-moon"></i> Dark Mode

    </button>

    <header>

        <h1><i class="fas fa-bell"></i> Notifications</h1>

    </header>

    <nav>

        <a href="dashboard.html"><i class="fas fa-home"></i> Dashboard</a>

        <a href="appointments.html"><i class="fas fa-calendar-check"></i> Appointments</a>

        <a href="patients.html"><i class="fas fa-user-injured"></i> Patients</a>

        <a href="doctors.html"><i class="fas fa-user-md"></i> Doctors</a>

    </nav>

    <div class="container">

        <div class="content">

            <h2>Upcoming Appointments</h2>

            <div class="filters">

                <select id="filter-time">

                    <option value="all">All Time Periods</option>

                    <option value="today">Today</option>

                    <option value="tomorrow">Tomorrow</option>

                    <option value="week">This Week</option>

                    <option value="month">This Month</option>

                </select>

                <select id="filter-doctor">

                    <option value="all">All Doctors</option>

                    <option value="dr-smith">Dr. Smith</option>

                    <option value="dr-johnson">Dr. Johnson</option>

                    <option value="dr-williams">Dr. Williams</option>

                </select>

                <button id="filter-button"><i class="fas fa-filter"></i> Apply Filters</button>

            </div>

            <input type="text" id="search-input" placeholder="Search by patient name or notification ID">

            <table id="notifications-table">

                <thead>

                    <tr>

                        <th>Notification ID</th>

                        <th>Patient Name</th>

                        <th>Doctor Name</th>

                        <th>Appointment Date</th>

                        <th>Appointment Time</th>

                        <th>Status</th>

                        <th>Actions</th>

                    </tr>

                </thead>

                <tbody id="notifications-body">

                    <!-- Data will be populated dynamically -->

                </tbody>

            </table>

        </div>

    </div>

    <footer>

        <p>&copy; 2025 Health Connect - Your Health, Our Priority</p>

    </footer>

    <script>

        let allAppointments = []; // Store all appointments for filtering

        function fetchAppointments() {

            fetch('/api/notifications/appointments')

                .then(response => response.json())

                .then(appointments => {

                    allAppointments = appointments; // Store all appointments

                    updateAppointmentsDisplay(appointments);

                    populateDoctorsFilter(appointments); // Populate doctors filter dynamically

                })

                .catch(error => {

                    console.error('Error:', error);

                    document.getElementById('notifications-body').innerHTML = `

                        <tr>

                            <td colspan="7" style="text-align: center; color: red;">

                                Error loading appointments. Please try again later.

                            </td>

                        </tr>

                    `;

                });

        }

        function updateAppointmentsDisplay(appointments) {

            const tbody = document.getElementById('notifications-body');

            tbody.innerHTML = '';

            if (appointments.length === 0) {

                tbody.innerHTML = `

                    <tr>

                        <td colspan="7" style="text-align: center;">No appointments found</td>

                    </tr>

                `;

                return;

            }

            appointments.forEach(appt => {

                const tr = document.createElement('tr');

                const statusClass = getStatusClass(appt.status);

                const statusText = getStatusText(appt.status);

                tr.innerHTML = `

                    <td>N${String(appt.notification\_id).padStart(3, '0')}</td>

                    <td><i class="fas fa-user-injured"></i> ${appt.patient\_name}</td>

                    <td><i class="fas fa-user-md"></i> ${appt.doctor\_name}</td>

                    <td>${appt.appointment\_date}</td>

                    <td>${appt.appointment\_time}</td>

                    <td><span class="badge ${statusClass}">${statusText}</span></td>

                    <td class="action-buttons">

                        <button class="notify-button" onclick="notifyPatient(${appt.notification\_id})">

                            <i class="fas fa-paper-plane"></i>

                        </button>

                        <button class="reschedule-button secondary" onclick="rescheduleAppointment(${appt.notification\_id})">

                            <i class="fas fa-calendar-alt"></i>

                        </button>

                    </td>

                `;

                tbody.appendChild(tr);

            });

        }

        function populateDoctorsFilter(appointments) {

            const doctorSelect = document.getElementById('filter-doctor');

            const doctors = [...new Set(appointments.map(appt => appt.doctor\_name))];

            doctorSelect.innerHTML = '<option value="all">All Doctors</option>';

            doctors.forEach(doctor => {

                doctorSelect.innerHTML += `<option value="${doctor}">${doctor}</option>`;

            });

        }

        function filterAppointments() {

            const timeFilter = document.getElementById('filter-time').value;

            const doctorFilter = document.getElementById('filter-doctor').value;

            const searchTerm = document.getElementById('search-input').value.toLowerCase();

            let filtered = allAppointments;

            // Apply time filter

            if (timeFilter !== 'all') {

                filtered = filtered.filter(appt => appt.status === timeFilter);

            }

            // Apply doctor filter

            if (doctorFilter !== 'all') {

                filtered = filtered.filter(appt => appt.doctor\_name === doctorFilter);

            }

            // Apply search filter

            if (searchTerm) {

                filtered = filtered.filter(appt =>

                    appt.patient\_name.toLowerCase().includes(searchTerm) ||

                    String(appt.notification\_id).includes(searchTerm)

                );

            }

            updateAppointmentsDisplay(filtered);

        }

        // Initialize page

        document.addEventListener('DOMContentLoaded', function() {

            fetchAppointments();

            // Add event listeners for filters

            document.getElementById('filter-time').addEventListener('change', filterAppointments);

            document.getElementById('filter-doctor').addEventListener('change', filterAppointments);

            document.getElementById('search-input').addEventListener('input', filterAppointments);

            // Set initial theme from localStorage

            const savedTheme = localStorage.getItem('theme') || 'light';

            document.documentElement.setAttribute('data-theme', savedTheme);

            // Update initial button state

            const themeToggle = document.querySelector('.theme-toggle');

            const icon = themeToggle.querySelector('i');

            icon.className = `fas ${savedTheme === 'dark' ? 'fa-sun' : 'fa-moon'}`;

            themeToggle.innerHTML = `${icon.outerHTML} ${savedTheme === 'dark' ? 'Light Mode' : 'Dark Mode'}`;

        });

        function getStatusClass(status) {

            switch (status) {

                case 'today':

                    return 'badge-today';

                case 'tomorrow':

                    return 'badge-upcoming';

                case 'week':

                    return 'badge-upcoming';

                default:

                    return 'badge-upcoming';

            }

        }

        function getStatusText(status) {

            switch (status) {

                case 'today':

                    return 'Today';

                case 'tomorrow':

                    return 'Tomorrow';

                case 'week':

                    return 'This Week';

                default:

                    return 'Upcoming';

            }

        }

        function notifyPatient(id) {

            // Implement notification logic here

            alert(`Notification sent for appointment #${id}`);

        }

        function rescheduleAppointment(id) {

            // Implement reschedule logic here

            alert(`Reschedule appointment #${id}`);

        }

        function toggleTheme() {

            const html = document.documentElement;

            const currentTheme = html.getAttribute('data-theme');

            const newTheme = currentTheme === 'dark' ? 'light' : 'dark';

            html.setAttribute('data-theme', newTheme);

            localStorage.setItem('theme', newTheme);

            // Update button icon and text

            const themeToggle = document.querySelector('.theme-toggle');

            const icon = themeToggle.querySelector('i');

            icon.className = `fas ${newTheme === 'dark' ? 'fa-sun' : 'fa-moon'}`;

            themeToggle.innerHTML = `${icon.outerHTML} ${newTheme === 'dark' ? 'Light Mode' : 'Dark Mode'}`;

        }

    </script>

</body>

</html>

**REFERENCES:**

1. **https://www.w3schools.com/Html/**
2. **https://github.com/kishan0725/Hospital-Management-System**
3. [**https://www.booknmeet.com/**](https://www.booknmeet.com/)