# NEW SUMMIT COLLEGE

**(Affiliated to Tribhuvan University)**

**Institute of Science and Technology**

**A Project Report**

# Doctor Appointment Booking System

*Submitted in Partial Fulfillment of the Requirement of*

**Bachelor of Computer Science and Information Technology**
**(BSc. CSIT)**

**<u>Submitted By:</u>**

Sneha Sah (5-2-476-35-2022)

27 April 2025

# ACKNOWLEDGEMENT

I would like to express my deepest gratitude to all those who have contributed to the successful completion of this project. First and foremost, I extend my sincere thanks to **Mr. Parichaya Adhikari**, my project guide, for their invaluable guidance, support, and encouragement throughout this project. Their insights and expertise have been crucial in shaping the direction and execution of the project. I am also grateful to **New Summit College** for providing the necessary resources and a conducive environment to carry out my research and development work. Special thanks to the doctors, hospital staff, and patients who participated in the user testing phase, providing valuable feedback that helped improve the system's functionality and user experience. I would like to acknowledge my peers and friends for their constructive suggestions and moral support during challenging times. Lastly, I am forever indebted to my family for their unwavering support, patience, and understanding throughout the project duration. Thank you all for your contributions to the success of this project.

**With Regards**

Sneha Sah

# ABSTRACT

The Doctor Appointment Booking System is designed to simplify and streamline the process of scheduling medical appointments for both patients and healthcare providers. This project addresses the common challenges associated with traditional appointment booking methods, such as long waiting times, manual record-keeping, and communication inefficiencies. The system provides an intuitive and user-friendly platform where patients can easily book, reschedule, or cancel appointments online. It allows doctors to manage their schedules efficiently, view patient details, and keep track of upcoming consultations. The system also includes administrative features for managing user accounts, doctors' availability, and generating reports. Developed using modern web technologies, this system ensures secure data handling, real-time updates, and a responsive design for accessibility on various devices. The implementation of robust security measures protects sensitive patient information, adhering to privacy regulations. The Doctor Appointment Booking System not only enhances the patient experience by reducing waiting times and improving convenience but also supports healthcare providers in optimizing their workflow. This project demonstrates how technology can be leveraged to improve the efficiency and effectiveness of healthcare services.

***Keywords:*** *Patients, Database Management, Doctor Management, Admin Management*

# Table of Contents

ACKNOWLEDGEMENT ..... ii

ABSTRACT ..... iii

List of Figure ..... vi

List of Table ..... vii

List of Abbreviations ..... viii

CHAPTER 1: INTRODUCTION ..... 1

1.1 Introduction ..... 1

1.2 Problem Statement ..... 1

1.3 Objectives ..... 1

1.4 Scope & Limitation ..... 2

1.4.1 Scope ..... 2

1.4.2 Limitation ..... 2

1.5 Development Methodology ..... 2

1.6 Report Organization ..... 3

CHAPTER 2: BACKGROUND STUDY & LITERATURE REVIEW ..... 5

2.1 Background Study ..... 5

2.2 Literature Review ..... 5

CHAPTER 3: SYSTEM ANALYSIS & DESIGN ..... 6

3.1 System Analysis ..... 6

3.1.1 Requirement Analysis ..... 6

3.1.2 Feasibility Analysis ..... 8

3.1.3 Object Modeling using Class & Object Diagram ..... 9

3.1.4 Dynamic Modeling using Sequence Diagram ..... 11

3.1.5 Process Modeling using Activity Diagram ..... 14

3. 2 System Design ..... 16

3.2.1 Refinement of Class, Object, State, Sequence and Activity Diagram ..... 16

3.2.2 Component Diagram ..... 16

CHAPTER 4: IMPLEMENTATION & TESTING ........ 17

4.1 Implementation ........ 17

4.1.1 Tools Used ........ 17

4.1.2 Implementation Detail of Modules ........ 17

4.2 Testing ........ 20

CHAPTER 5: CONCLUSION & FUTURE RECOMMENDATION ........ 23

5.1 Conclusion ........ 23

5.2 Future Recommendation ........ 23

REFERENCES ........ 24

APPENDICS ........ 25

# List of Figure

Figure 1.1 Waterfall Model ....................3
Figure 3.1 Use Case Diagram....................7
Figure 3.2 Class Diagram....................10
Figure 3.3 Object Diagram....................11
Figure 3.4 Admin Sequence Diagram ....................12
Figure 3.5 Doctor Sequence Diagram ....................13
Figure 3.6 Patient Sequence Diagram ....................14
Figure 3.7 Activity Diagram....................15
Figure 3.8 Component Diagram....................17

# List of Table

Table 3.1 Gantt Chart ........................................................................................................ 9

# List of Abbreviations

| | |
|---|---|
| CSS | Cascading Style Sheet |
| HTML | Hypertext Markup Language |
| JS | Java Script |
| SDLC | Software Development Life Cycle |
| SQL | Structure Query Language |
| VS CODE | Visual Studio Code |

# CHAPTER 1: INTRODUCTION

## 1.1 Introduction

In today's fast-paced world, the healthcare sector is continually evolving to meet the growing needs of patients. One of the critical challenges faced by healthcare providers is the efficient management of patient appointments. Traditional methods of booking appointments, which often involve phone calls or in-person visits, can be time-consuming and prone to errors. These methods also lead to inefficiencies such as double bookings, missed appointments, and long waiting times. To address these challenges, the Doctor Appointment Booking System has been developed as a modern solution that leverages technology to streamline the appointment scheduling process. This system provides a platform where patients can easily book, reschedule, or cancel appointments through a web based or mobile interface, reducing the need for physical visits and manual processes. For doctors and healthcare providers, the system offers tools to manage their schedules effectively, monitor patient appointments, and ensure optimal utilization of their time. The system also supports administrators in managing doctor availability, patient records, and generating insightful reports for operational improvements.

## 1.2 Problem Statement

Conventional approaches to scheduling doctor's visits, such phone calls or walk-ins, are ineffective and prone to mistakes like missed appointments and duplicate reservations. The entire quality of service is impacted by these procedures, which result in lengthy waiting times for patients and scheduling difficulties for medical professionals.

An automated system that streamlines appointment scheduling, lowers errors, and boosts productivity is required. By offering a simplified, user-friendly platform for consumers and healthcare providers, the Doctor Appointment Booking System seeks to address these problems.

## 1.3 Objectives

- To develop a platform for booking, rescheduling, or canceling doctor appointments online.
- To enhance the efficiency of healthcare providers in managing their schedules.

- To reduce errors and inefficiencies associated with manual appointment booking methods.

## 1.4 Scope & Limitation

### 1.4.1 Scope

The scope of this project is to create an efficient, user-friendly system for booking doctor appointments. It aims to improve patient experience and streamline healthcare providers' schedules.

- Accessible through web and mobile interfaces.
- Supports real-time updates for appointments and availability.
- Implements secure data handling practices to protect sensitive patient information.
- It will support real-time updates for appointments and availability.

### 1.4.2 Limitation

- The system requires an internet connection for real-time updates.
- Initial setup and training may be needed for users unfamiliar with digital platforms.

## 1.5 Development Methodology

Waterfall approach was the first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases [1].

The project will adopt the waterfall methodology, a sequential approach where each phase must be completed before the next beginning. This ensures thorough documentation and a structured development process, starting with requirements gathering and moving through design, implementation, testing, deployment, and maintenance.

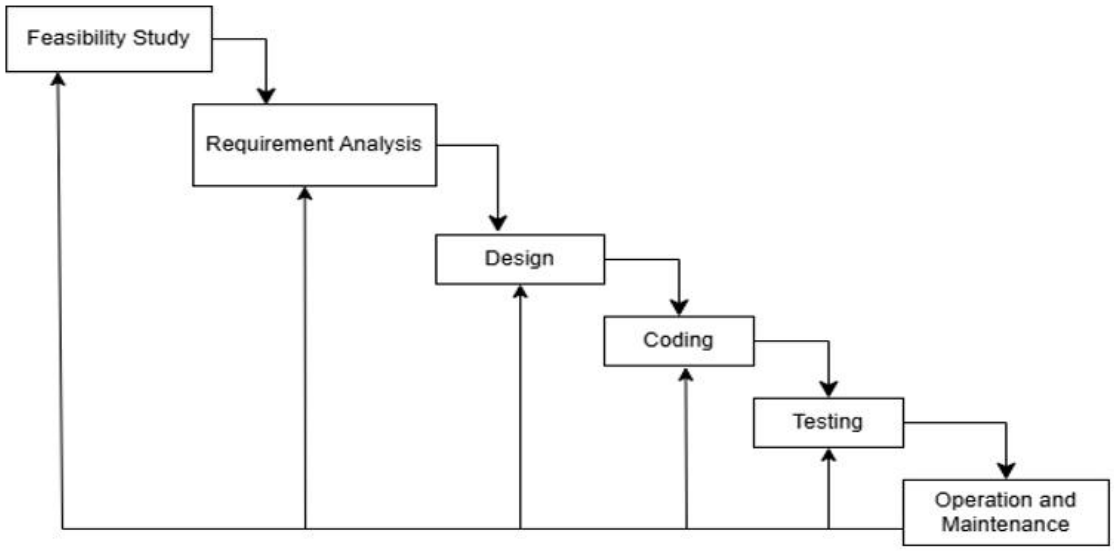


*Figure 1.1 Waterfall Model*

- **Requirements Analysis:** Understand and document all the features and functionalities the software needs to have.
- **System Design:** Plan the overall structure of the software, including hardware needs and how different parts will work together.
- **Implementation:** Build the software in small, testable pieces (units), ensuring each part functions correctly.
- **Testing:** Combine all the units into a complete system and thoroughly test it for any errors.
- **Deployment:** Release the finished software to the customer or the market.
- **Maintenance:** Ongoing support after release, including fixing issues (patches) and improving the software with new versions [1].

## 1.6 Report Organization

The project is organized into five distinct chapters, each representing a different stage of development. A summary of these chapters is as follows:

- **Chapter 1: "Introduction"** – Introduces the project by outlining its objectives, purpose, and scope. It also describes the project's overall plan to achieve its goals while maintaining alignment with its core concept and development approach.
- **Chapter 2: "Background Study and Literature Review"** – Covers the preliminary research and preparation conducted before project initiation. This chapter includes a

detailed literature review to analyze similar projects, identify gaps, and establish a strong foundation for improvement.

- **Chapter 3: "System Analysis and Design"** – Focuses on analyzing the project's functional and non-functional requirements, along with feasibility studies from economic, technical, and operational perspectives. Additionally, it details the system's design, including processes, databases, and user interfaces, based on the project's conceptual framework.
- **Chapter 4: "Implementation and Testing"** – Discusses the system's development and testing phase. It explains the tools and techniques used to implement the system design and ensure its functionality and reliability.
- **Chapter 5: "Conclusion and Future Recommendations"** – Concludes the project by summarizing key insights, lessons learned, and recommendations for future enhancements. It emphasizes improving the system's functionality and user experience while offering guidance for future projects.

# CHAPTER 2: BACKGROUND STUDY & LITERATURE REVIEW

## 2.1 Background Study

The healthcare sector is increasingly adopting digital solutions to improve service delivery. Automated appointment booking systems have emerged as a significant innovation in this domain, helping to reduce administrative burdens and enhance patient satisfaction. According to a study by Smith published in the Journal of Healthcare Information Management, digital booking systems improve efficiency by minimizing human errors and reducing the workload on administrative staff [2].

Moreover, a survey conducted by the International Journal of Medical Informatics emphasizes that digital systems enhance patient engagement by providing greater control over appointment scheduling, which ultimately leads to higher satisfaction levels. These findings underscore the potential of digital solutions like the Doctor Appointment Booking System to revolutionize traditional healthcare practices by introducing more efficiency and accuracy [3].

## 2.2 Literature Review

Various appointment booking systems have been developed, each with unique features and limitations. For example, some systems offer basic functionality such as appointment scheduling and reminders, while others integrate advanced features like patient record management and analytics. A study by [4] in the Journal of Medical Systems highlighted that most existing systems lack comprehensive integration with healthcare providers' existing IT infrastructure, leading to inefficiencies [4]. Additionally, Smith and Lee in their book Digital Healthcare Solutions pointed out that many systems fail to address user experience adequately, often resulting in patient dissatisfaction. Existing appointment booking systems will be evaluated for strengths like ease of use and integration, and weaknesses such as limited customization and inadequate security. Some systems offer seamless scheduling but often fall short in data protection, as noted by the Journal of Digital Health (2019). This project aims to develop a Doctor Appointment Booking System that addresses these gaps by providing a secure, customizable, and efficient platform, enhancing healthcare service delivery [5].

# CHAPTER 3: SYSTEM ANALYSIS & DESIGN

## 3.1 System Analysis

The Doctor Appointment Booking System is designed to manage the scheduling of doctor appointments for patients. The system will be used by patients, doctors, and administrative staff. It will allow patients to book, cancel, or reschedule appointments online, while doctors can view and manage their schedules. Administrative staff will manage patient information and appointment schedules. The system will also generate reminders and notifications for upcoming appointments.

The system will be available 24/7 and respond to user requests within 2 seconds, ensuring data security and integrity. It will be scalable to accommodate increasing user traffic. Patients will interact with the system through a user interface, while doctors and administrative staff will use the system to manage their schedules and patient information. The system will also interact with external systems, such as payment gateways and electronic health records, to facilitate seamless integration.

### 3.1.1 Requirement Analysis

The Doctor Appointment Booking System requires several functional and non-functional requirements to ensure its effectiveness. Functional requirements include user authentication, appointment booking, rescheduling, cancellation, and notification. These features will enable patients to manage their appointments and receive timely reminders. Non-functional requirements include system reliability, security, scalability, and usability, ensuring that the system is available 24/7, secure, and easy to use.

#### 3.1.1.1 Functional Requirements

The Doctor Appointment Booking System must provide a comprehensive set of functional capabilities to meet the needs of patients. Key functional requirements include secure user authentication, allowing patients to create accounts and log in securely. The system should also offer a user-friendly interface for booking, rescheduling, and canceling appointments, as well as sending timely notifications to patients about their scheduled visits.

**User:**

- Create and manage personal profiles and access through a secure login.
- Book and oversee their appointments.
- Track their appointment details and status updates.

**Doctor:**

- Access the system through login credentials.
- Review scheduled appointments.

**Admin:**

- Access the system using login credentials.
- Add and manage admin accounts.
- Register new doctors.
- Handle appointments and messages.

**Use Case Diagram**

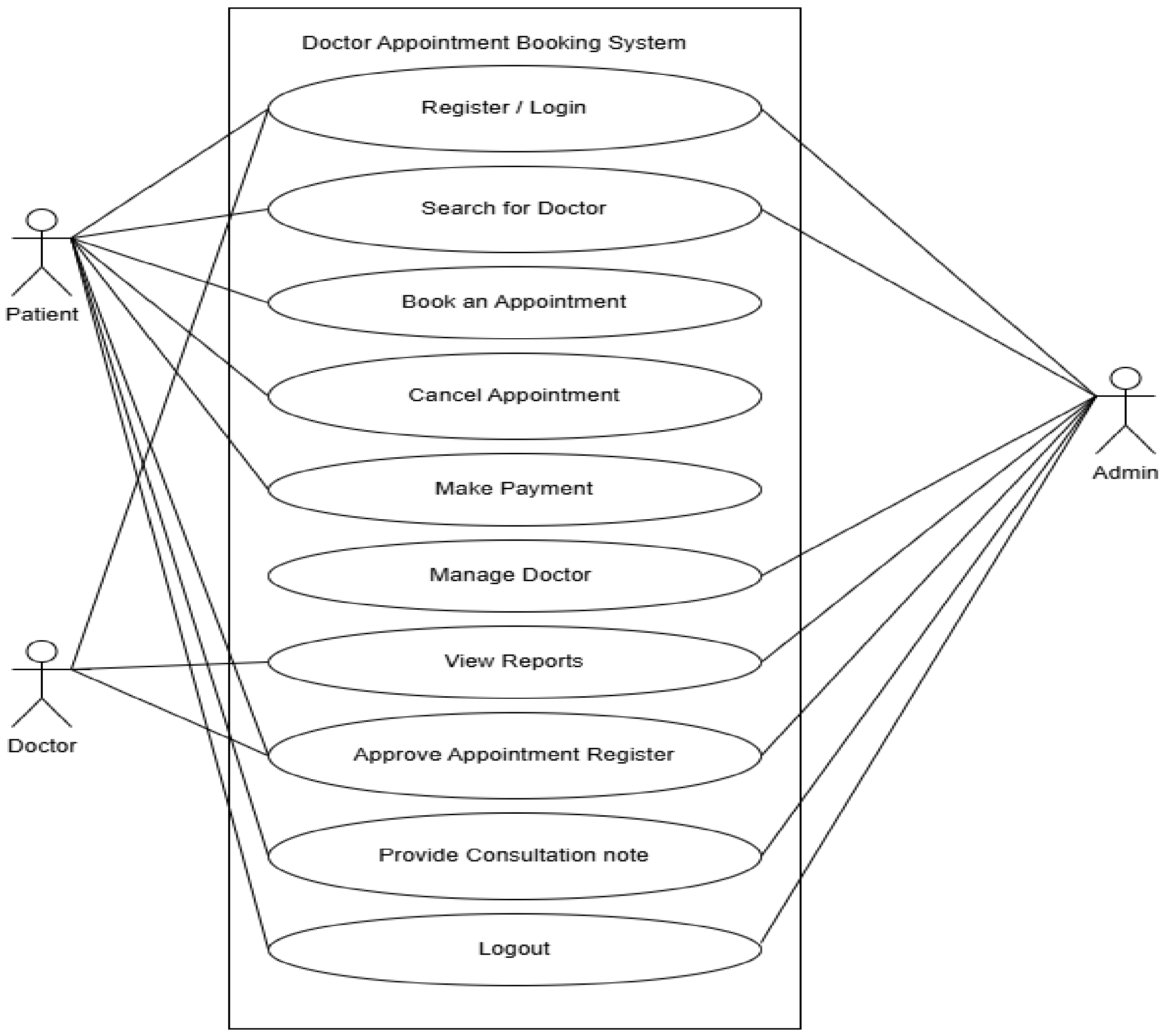


*Figure 3.1 Use Case Diagram*

The use case diagram illustrates three primary actors: Admin, User, and Doctor. It includes four distinct use cases, each representing specific functionalities within the Doctor Appointment System. Each actor interacts with relevant use cases based on their role.

The User (or patient) can register, log in, create appointments, send messages, view their own appointments, manage their profile, and log out. The Doctor is limited to logging in and viewing appointments assigned specifically to them. The admin, as the third actor, can log in, add new doctors and admins, view and manage appointments, handle messages, manage users, and log out. The interactions of these three actors collectively define the core functionalities of the Doctor Appointment System.

#### 3.1.1.2 Non-Functional Requirements

The Doctor Appointment Booking System must ensure high reliability and availability with minimal downtime, along with strong security measures to protect patient data. It should be scalable to accommodate growing user demand without performance issues. Additionally, the system must be user-friendly with a responsive web design, providing an intuitive interface for seamless appointment management.

### 3.1.2 Feasibility Analysis

Feasibility analysis is a crucial step in evaluating the practicality and viability of the Doctor Appointment Booking System. This assessment ensures that the system can be successfully developed, implemented, and maintained, while meeting the technical, operational, and economic requirements.

**1. Technical Feasibility**

The system is technically feasible as it utilizes readily available technologies and tools for development. Modern programming languages, frameworks, and database systems provide the foundation for creating a responsive, secure, and scalable web application. Additionally, existing IT infrastructure can support the deployment and maintenance of the system without major upgrades.

**2. Operational Feasibility**

The operational feasibility of the system lies in its user-friendly design, which ensures easy navigation for patients, doctors, and administrative staff. By providing an intuitive interface, the system simplifies appointment booking and management processes. Training requirements for end-users are minimal, as the system is designed with accessibility and efficiency in mind, enhancing overall satisfaction.

**3. Economic Feasibility**

From an economic perspective, the system proves to be cost-effective. A cost-benefit analysis shows that investment in system development will lead to reduced administrative expenses, increased efficiency, and improved patient satisfaction. The long-term benefits outweigh the initial costs, making the implementation a financially sound decision.

**4. Schedule**

The completion of this report follows a well-structured timeline to ensure all topics are thoroughly covered and organized. A detailed Gantt chart has been prepared to outline the schedule, specifying the days allocated to each section of the report. This systematic approach ensures timely progress and allows for a comprehensive, well-prepared final document.

| Working Time | 15th Feb | 21th Feb | 5th March | 25th March | 3rd April | 10th April |
|---|---|---|---|---|---|---|
| Requirement Analysis | ■ | | | | | |
| System Design | | ■ | | | | |
| Coding | | | ■ | | | |
| Implementation | | | | ■ | | |
| Testing | | | | | ■ | ■ |
| Documentation | ■ | ■ | ■ | ■ | ■ | ■ |

*Table 1 Gantt Chart*

### 3.1.3 Object Modeling using Class & Object Diagram

Object modeling is essential for structuring and visualizing the Doctor Appointment Booking System. Using Class and Object Diagrams, the system is represented by defining key classes, their attributes, methods, and relationships. The Class Diagram serves as a blueprint, outlining primary entities such as Admin, Patient, Doctor, Appointment, Prescription, and Payment. The Patient class includes attributes like patientID, name, and email, with methods such as register() and bookAppointment(). The Doctor class consists of doctorID, specialization, and availability(), with functions like addAvailability() and viewAppointments(). The Appointment class manages scheduling, cancellations, and updates. Additionally, doctors can generate Prescriptions, which are linked to patients, while the Admin oversees system management, including user approvals and reporting. The Payment class handles billing for consultations. Relationships between these classes define how patients book appointments, doctors manage schedules, and the system ensures smooth interactions between users.

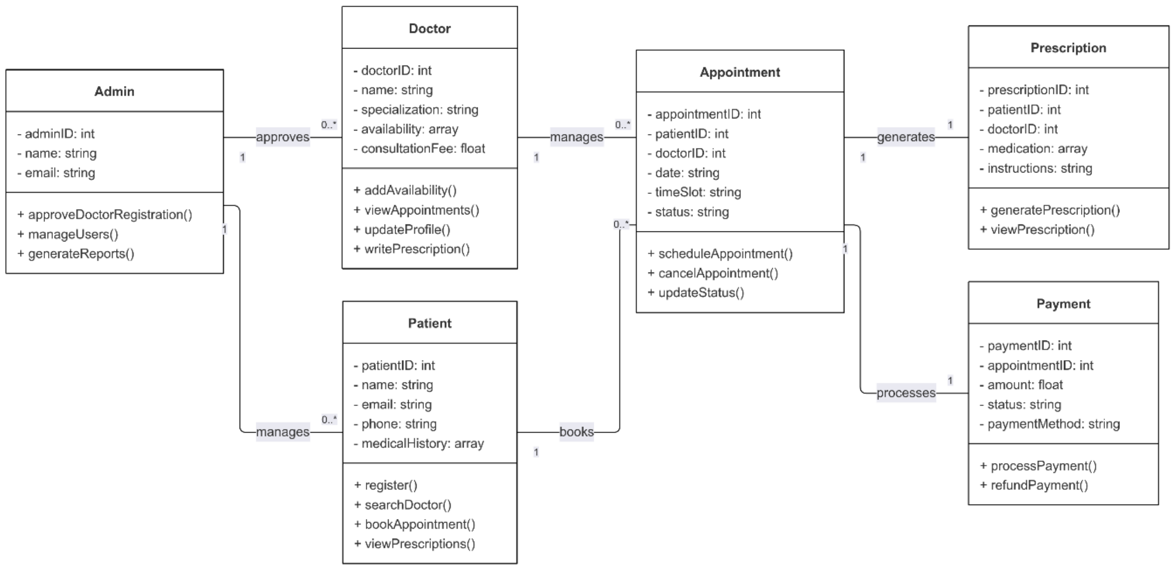


*Figure 2.2 Class Diagram*

An Object Diagram represents specific instances of classes at a given moment, showcasing how objects interact in a real-world scenario. Below is an example object diagram for a Doctor Appointment Booking System, illustrating instances of Admin, Patient, Doctor, Appointment, Prescription, and Payment.

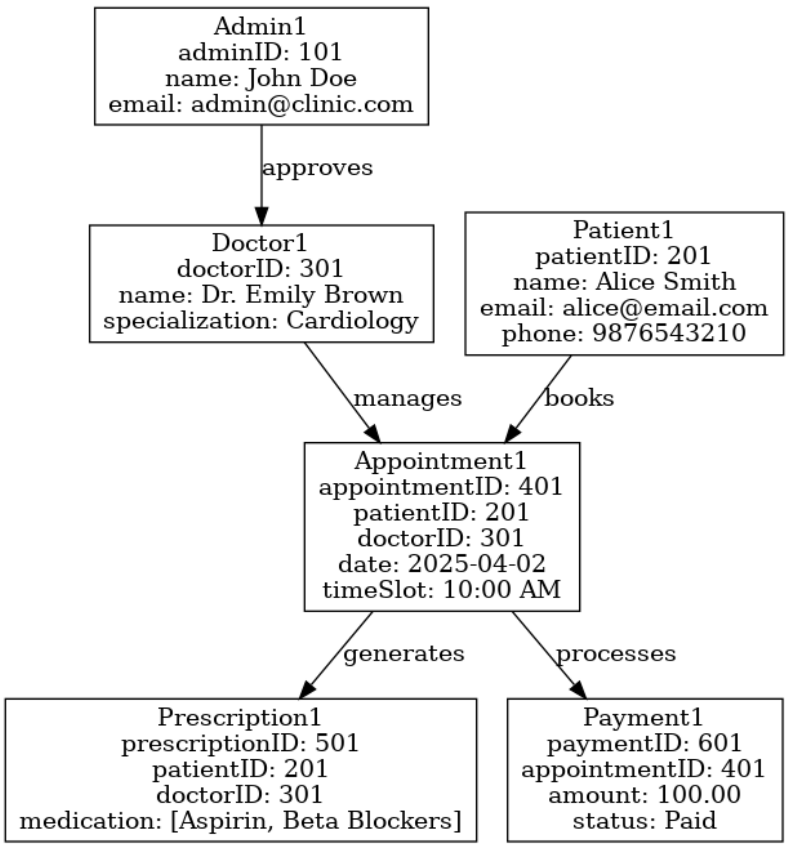


*Figure 3.3 Object Diagram*

### 3.1.4 Dynamic Modeling using Sequence Diagram

Dynamic modeling illustrates how the Doctor Appointment Booking System behaves during runtime, focusing on state changes and object interactions. It includes State Diagrams and Sequence Diagrams to represent different processes.

The Sequence Diagram showcases the interaction between entities—Patient, System, Doctor, and Payment Module, demonstrating the process flow. A patient searches for a doctor, books an appointment, and the system notifies the doctor. Upon confirmation, payment is processed, and the doctor conducts the consultation, providing a prescription. These diagrams help visualize the system's dynamic behavior and communication flow efficiently.

**Admin Sequence Diagram**

An admin sequence diagram for a doctor appointment system would depict the administrative tasks, such as managing doctor profiles, approving or rescheduling appointments, and handling system configurations. It would show the admin interacting with the system to perform these actions, often involving database interactions for data manipulation and system updates.

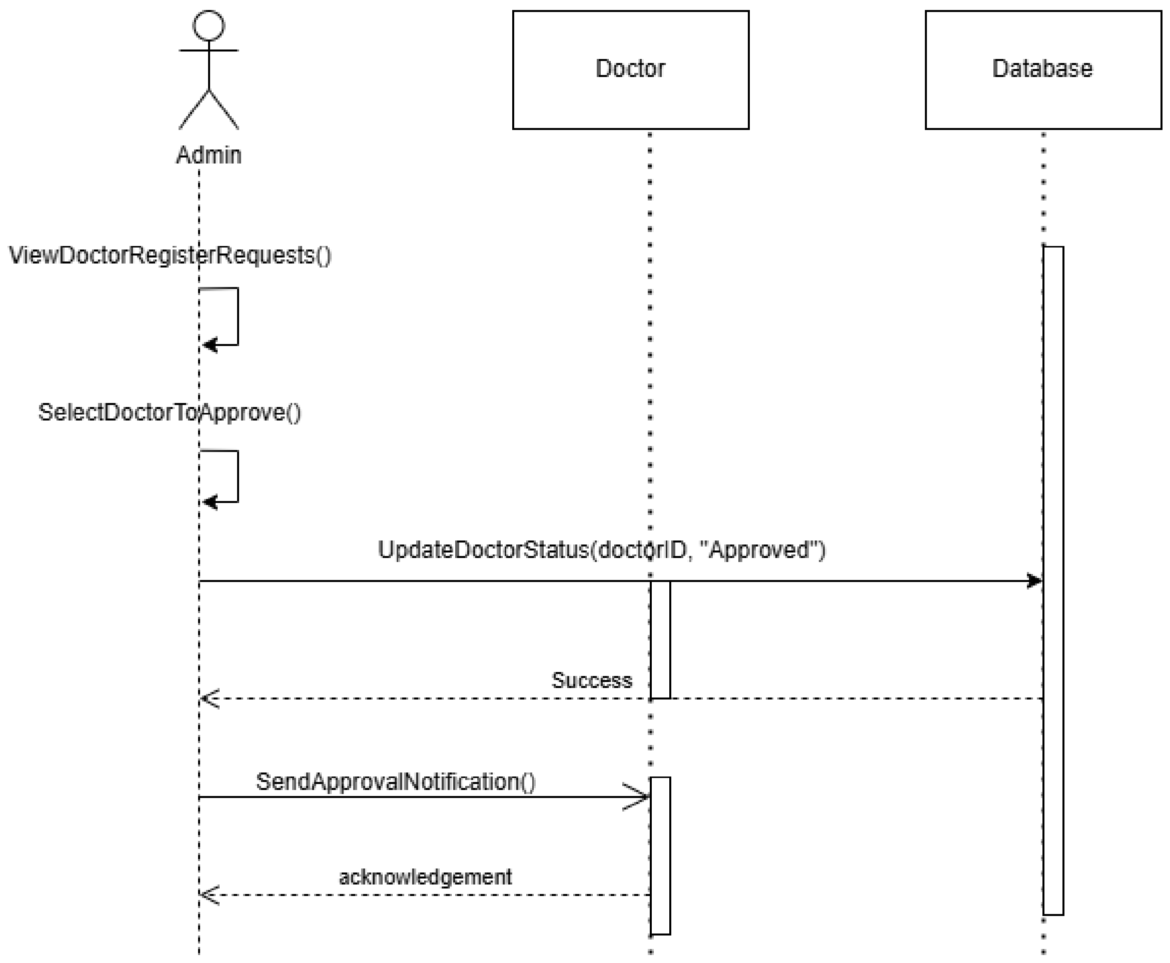

Admin
Doctor
Database
ViewDoctorRegisterRequests()
SelectDoctorToApprove()
UpdateDoctorStatus(doctorID, "Approved")
Success
SendApprovalNotification()
acknowledgement

*Figure 3.4 Admin Sequence Diagram*

### Doctor Sequence Diagram

A doctor's sequence diagram focuses on the interactions a doctor has with the appointment system. It typically includes scenarios like viewing their daily schedule, confirming or rescheduling appointments, accessing patient records, and potentially updating their availability. It highlights communication between the doctor, the system, and the database to manage their appointments and patient information.

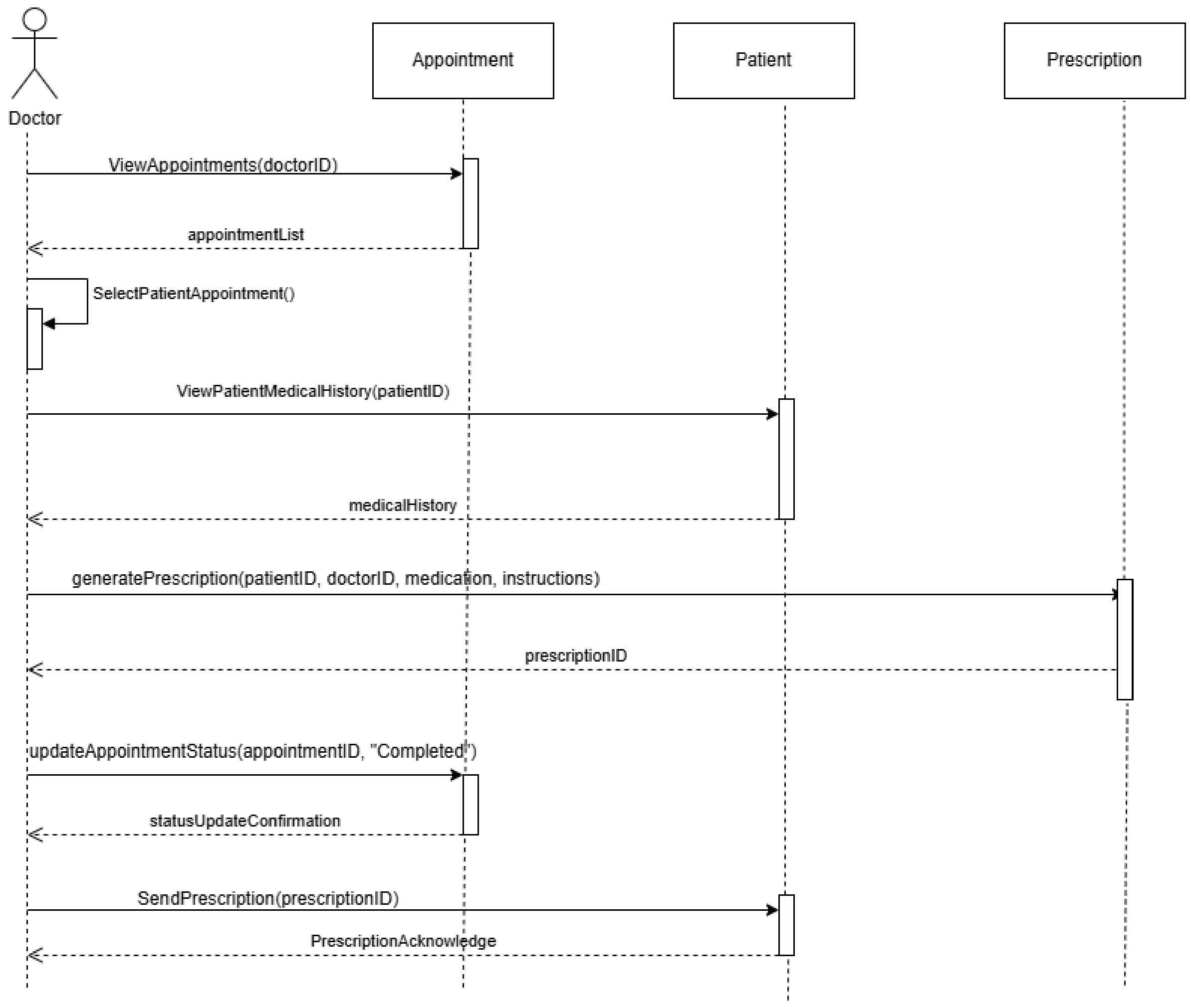

Doctor
Appointment
Patient
Prescription
ViewAppointments(doctorID)
appointmentList
SelectPatientAppointment()
ViewPatientMedicalHistory(patientID)
medicalHistory
generatePrescription(patientID, doctorID, medication, instructions)
prescriptionID
updateAppointmentStatus(appointmentID, "Completed")
statusUpdateConfirmation
SendPrescription(prescriptionID)
PrescriptionAcknowledge

*Figure 3.5 Doctor Sequence Diagram*

## Patient Sequence Diagram

A patient sequence diagram for a doctor appointment system outlines the patient's interactions with the system. Common scenarios include searching for doctors, booking appointments, viewing appointment details, canceling appointments, and potentially accessing medical records or communicating with the doctor. The diagram emphasizes the flow of information between the patient, the system, and the database, showing how the patient interacts with the system to manage their healthcare appointments.

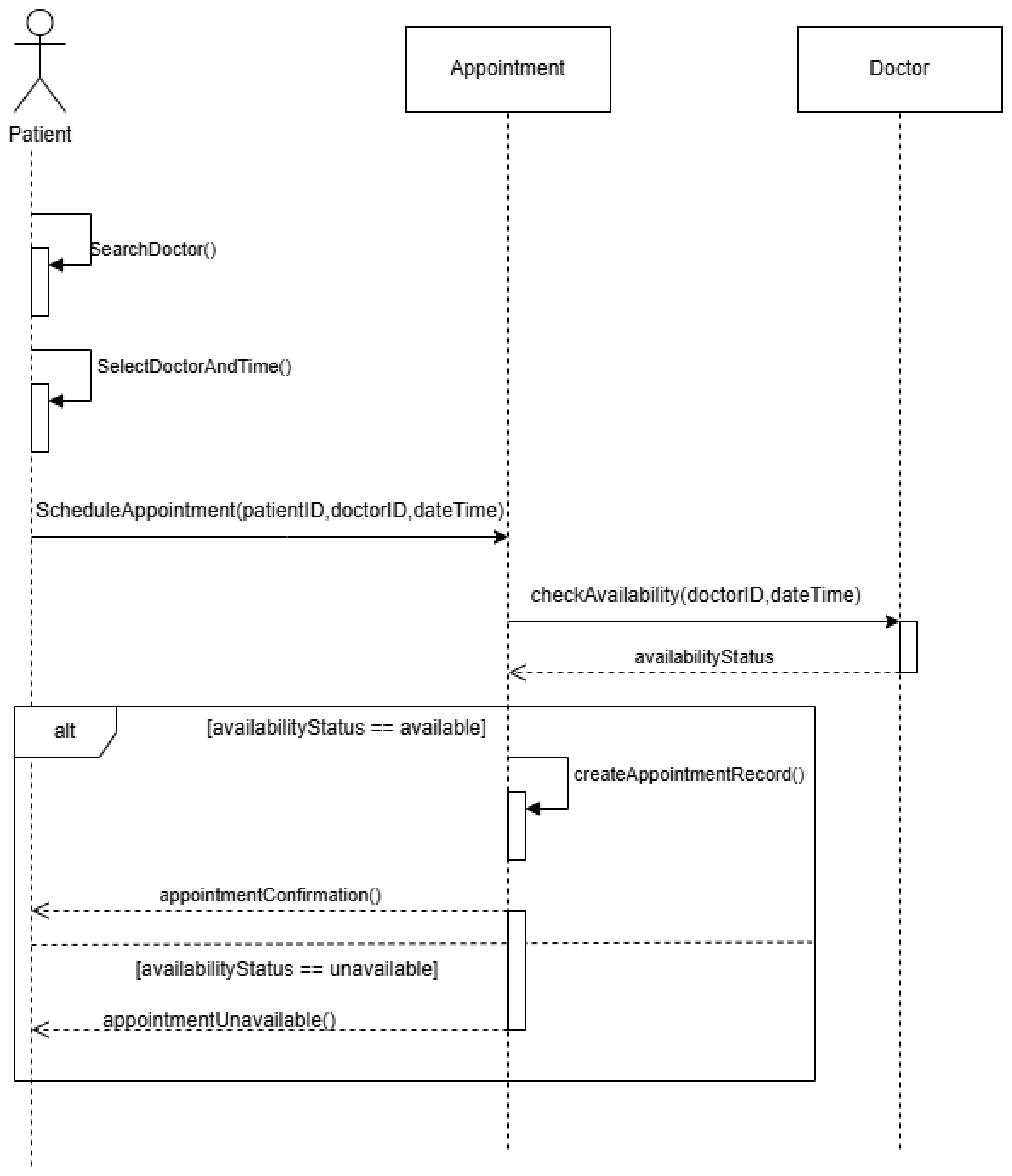

Patient
Appointment
Doctor
SearchDoctor()
SelectDoctorAndTime()
ScheduleAppointment(patientID,doctorID,dateTime)
checkAvailability(doctorID,dateTime)
availabilityStatus
alt
[availabilityStatus == available]
createAppointmentRecord()
appointmentConfirmation()
[availabilityStatus == unavailable]
appointmentUnavailable()

*Figure 3.6 Patient Sequence Diagram*

### 3.1.5 Process Modeling using Activity Diagram

The Activity Diagram for the Doctor Appointment Booking System illustrates the step-by-step process of booking an appointment. It begins with the Patient logging in, followed by searching for a Doctor based on specialization or availability. After selecting a doctor, the patient views details and proceeds to book an appointment. The request is sent to the doctor, who either approves or rejects it. If approved, the patient proceeds with payment, after which the system confirms the booking. On the scheduled date, the doctor conducts the consultation, and the process concludes. If the request is rejected, the process ends without an appointment. This diagram effectively maps out the system's workflow, decisions, and interactions involved in scheduling a medical consultation.

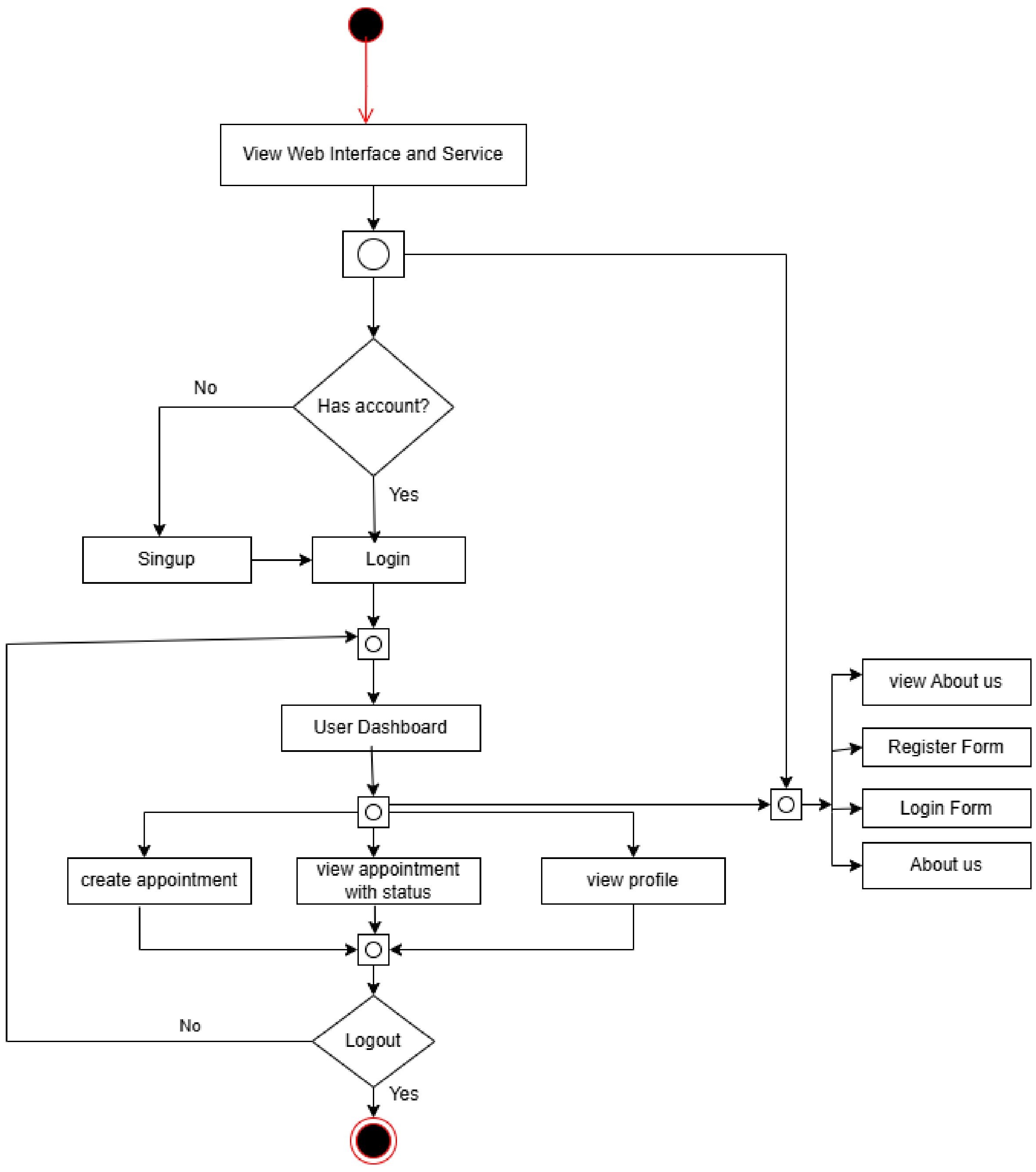


*Figure 3.7 Activity Diagram*

## 3. 2 System Design

The Doctor Appointment Booking System is designed to efficiently manage doctor-patient interactions, appointment scheduling, and administrative tasks. The system consists of three main modules: Patients, Doctors, and Admins. The backend database stores user profiles, appointment details, and payment records, ensuring secure and efficient data management. Technologies like Firebase or a relational database (MySQL/PostgreSQL) handle authentication and real-time updates. An intuitive frontend built with React.js ensures seamless navigation for users. The system ensures smooth interactions between modules, secure transactions, and automated appointment management.

### 3.2.1 Refinement of Class, Object, State, Sequence and Activity Diagram

Refinement involves adding more detail and complexity to diagrams as the system evolves, allowing better representation of functionality and behavior.

- **Class Diagram:** Initially, it defines basic entities like Patient and Doctor. Refinement adds attributes like Specialization and methods such as WritePrescription() and ViewAppointments(), making it more detailed.
- **Object Diagram:** Starts with simple instances like a specific Doctor object. Refinement adds Consultation Fee, Availability Slots, and Patient Medical History, improving clarity.
- **State Diagram:** Initially, it includes basic states like Available and Booked. Refinement introduces states like PendingApproval, Completed, and Cancelled, making transitions clearer.
- **Sequence Diagram:** Initially models interactions like Booking an Appointment. Refinement adds Payment Processing, Doctor Confirmation, and Prescription Generation, ensuring completeness.
- **Activity Diagram:** Begins with simple workflows like Searching for a Doctor. Refinement includes extra decision points such as Insurance Verification, Doctor Availability Check, and Payment Confirmation, enhancing user experience.

### 3.2.2 Component Diagram

A Component Diagram illustrates the core parts (components) of a system and their interactions. In the Doctor Appointment Booking System, key components like the Patient Module, Doctor Module, Admin Module, Appointment Scheduling Service, Notification Service, and Database work together to deliver the system's functionality.

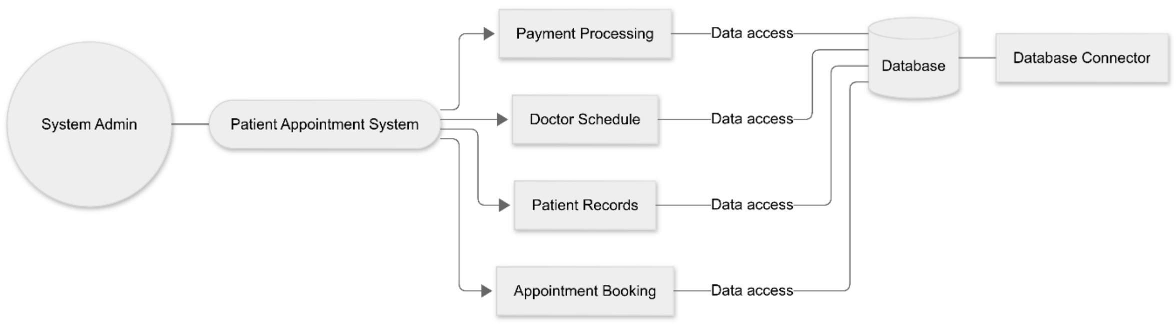


*Figure 3.8 Component Diagram*

# CHAPTER 4: IMPLEMENTATION & TESTING

## 4.1 Implementation

The Doctor Appointment Booking System was implemented using a combination of technologies to ensure a robust and scalable solution.

### 4.1.1 Tools Used

The implementation of the system involved the use of various tools and technologies, including:

- **Frontend:** Developed with HTML, CSS, and JavaScript for an intuitive and responsive user interface for all user roles (patients, doctors, and staff).
- **Backend:** Built on Node.js and Express for a scalable and efficient server-side, enabling a robust API.
- **Database:** Leveraged MySQL for secure and reliable data storage of patient, doctor, and appointment information.
- **Tools:** Utilized Visual Studio Code for development, Git for version control, and Postman for API testing.

### 4.1.2 Implementation Detail of Modules

Once the system design was finalized and any concerns addressed, the development phase began. While this document does not provide a complete implementation due to resource constraints, it outlines the primary components of the system. Below are some key modules of the Doctor Appointment Booking System:

**Admin Module**

The Admin is responsible for managing the system and overseeing doctor and patient interactions. Key functionalities include:

- approveDoctorRegistration(): Reviews and approves doctor profiles before they are listed.
- generateReports(): Analyzes system data, such as appointment trends and user activity.
- manageUsers(): Handles user accounts, including patients and doctors.

**Doctor Module**

This module allows doctors to manage their appointments and availability. The Doctor Class includes attributes like doctorID, name, specialization, and availability []. Key methods include:

- addAvailability(): Allows doctors to set available time slots.
- viewAppointments(): Displays scheduled patient bookings.
- updateProfile(): Enables doctors to modify their details, such as consultation fees and working hours.

**Patient Module**

Patients use this module to search for doctors and book appointments. The Patient Class includes attributes like userID, name, email, and medicalHistory[]. Important methods include:

- searchDoctors(): Finds doctors based on specialization, location, or availability.
- bookAppointment(): Schedules an appointment with a selected doctor.
- viewPrescriptions(): Allows patients to review prescriptions and past appointments.

**Procedures**

- **Search Procedure**
  Patients can find doctors using filters like specialization, location, and availability. When a user searches, the procedure searchDoctors(query, filters) retrieves relevant doctor profiles and presents them based on the selected criteria.
- **Appointment Booking Procedure**
  To schedule an appointment, the function bookAppointment(patientID, doctorID, timeSlot) is used. It verifies doctor availability before confirming the booking and sending notifications to both the doctor and the patient.
- **Prescription & Medical History Procedure**
  Doctors can generate prescriptions for patients using addPrescription(patientID, details), which stores the prescription in the database. Patients can then access it via viewPrescriptions(patientID).

**Functions**

- **Authentication Function**
  Handles user login and security. The function authenticateUser(email, password) checks user credentials and grants access if the details are correct.
- **Notification Function**
  Notifies users about upcoming appointments, cancellations, or prescription updates using sendNotification(userID, message).

**Methods**

Each class in the system includes methods for key actions:

- **Patient Class**
  - searchDoctors(): Finds doctors based on specialization and location.
  - bookAppointment(): Reserves an appointment slot with a selected doctor.
- **Doctor Class**
  - addAvailability(): Sets working hours for consultations.
  - manageAppointments(): Views and modifies scheduled appointments.

- **Admin Class**
    - approveDoctorRegistration(): Reviews and approves doctor profiles before activation.
    - generateReports(): Compiles system usage statistics.

**Algorithms**

- **Search Algorithm**
  Uses indexing and filtering to match user queries with doctor profiles. When a patient enters a search term, the algorithm finds the most relevant doctors based on specialization, location, and availability.

- **Appointment Scheduling Algorithm**
  Ensures that bookings are efficiently managed by checking available slots in real time before confirming an appointment.
- **Recommendation Algorithm**
  Based on previous research and appointment history, the system suggests doctors that best match the patient's preferences.

**Virtual Consultation Feature**

A Panorama View Feature allows patients to explore clinic interiors virtually before booking an appointment. This enhances user experience by providing a realistic preview of the facility. The implementation can utilize Google Street View Panorama API to embed interactive 360-degree views.

## 4.2 Testing

Testing is a critical phase conducted after the completion of the Doctor Appointment Booking System to ensure its functionality, reliability, and performance. Various types of testing are employed, including functional testing to verify that all features work as intended, usability testing to assess the system's user-friendliness, and security testing to safeguard sensitive patient data. Additionally, performance testing ensures the system can handle high user demand, and compatibility testing checks its responsiveness across different devices and browsers. These testing methodologies collectively ensure the system meets quality standards before deployment.

**Unit Testing**

| **Test Case ID** | **Module/ Function** | **Test Scenario** | **Test Steps** | **Expected Result** | **Status (Pass/Fail)** |
|---|---|---|---|---|---|
| 01 | Login Function | Valid login credentials | Enter valid username & password | User logged in successfully | Pass |
| 02 | Login Function | Invalid login credentials | Enter wrong password | Show "Invalid credentials" error | Pass |
| 03 | Search Doctor | Search by specialization | Enter "Dermatologist" in search | List of Dermatologists shown | Pass |
| 04 | Book Appointment | Book available slot | Select Dr. Sharma, date 1 May 2025, 10 AM slot | Appointment confirmation received | Pass |
| 05 | Book Appointment | Double booking prevention | Try to book some slot again | Show "Slot not available" error | Pass |
| 06 | Cancel Appointment | Valid appointment cancellation | Cancel booked appointment | Show "Appointment canceled" message | Pass |

| | | | | | |
|---|---|---|---|---|---|
| 07 | Payment Module | Successful payment | Enter valid payment details | Payment successfully, receipt generated | Pass |
| 08 | Payment Module | Failed payment | Enter invalid card details | Payment failed, retry option shown | Pass |

**System Testing**

| **Test Case ID** | **Test Scenario** | **Test Steps** | **Expected Result** | **Status (Pass/Fail)** |
|---|---|---|---|---|
| ST01 | New user registration & login | Register new user, login | User registered and logged in | Pass |
| ST02 | Full appointment booking flow | Login → Search doctor → Book slot → Payment | Booking confirmed, payment receipt received | Pass |
| ST03 | Cancel appointment after booking | Book appointment → Cancel it | Appointment canceled successfully | Pass |
| ST04 | Handle unavailable slots | Search doctor → Try booking a booked slot | "Slot not available" error shown | Pass |
| ST05 | Profile Management | Login → Update profile details → Save | Profile updated successfully | Pass |
| ST06 | Payment failure handling | Book appointment → Fail payment → Retry | Payment retries successful or booking canceled after limit | Pass |
| ST07 | Appointment History | Login → View past & upcoming appointments | List shown correctly | Pass |
| ST08 | Doctor Availability Update | Doctor updates availability → User searches | Updated slots reflected for users | Pass |

# CHAPTER 5: CONCLUSION & FUTURE RECOMMENDATION

## 5.1 Conclusion

The Doctor Appointment Booking System has been successfully implemented, providing a comprehensive solution for appointment booking and management. The system's user-friendly interface and robust backend have enabled patients to easily book, cancel, and reschedule appointments, while healthcare providers can efficiently manage their schedules and patient information. The system's implementation has demonstrated the potential for technology to improve healthcare services and patient outcomes, enhancing patient convenience and supporting healthcare providers in managing their schedules more effectively. The successful implementation of the Doctor Appointment Booking System has also highlighted the importance of considering the needs and requirements of all stakeholders involved in the healthcare process. By providing a platform that is accessible, user-friendly, and secure, the system has improved the overall experience for patients, doctors, and administrative staff.

## 5.2 Future Recommendation

While the Doctor Appointment Booking System provides a comprehensive solution for appointment booking, there are opportunities for future enhancements to further improve its functionality and scope. Some potential recommendations for future development include:

- Integrating Telemedicine Features: Integrating telemedicine features would enable patients to consult with doctors remotely, reducing the need for in-person visits and expanding access to healthcare services.
- Adding Support for Multiple Languages: Adding support for multiple languages would enable the system to cater to a broader range of patients, improving accessibility and usability.
- Expanding the System to Support Other Healthcare Services: Expanding the system to support other healthcare services, such as billing and insurance claims, would provide a more comprehensive solution for healthcare providers and patients.

# REFERENCES

[1] Tutorialpoints, "Tutorialpoints," [Online]. Available: https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm. [Accessed 15 Jan 2025].

[2] S. A. S. a. C. C. H. Mia Liza A. Lustria, "An examination of eHealth technology use in health information seeking, communication and personal health information management in the USA," *Health Informatics Journal,* no. 224-243, p. 09, 2011.

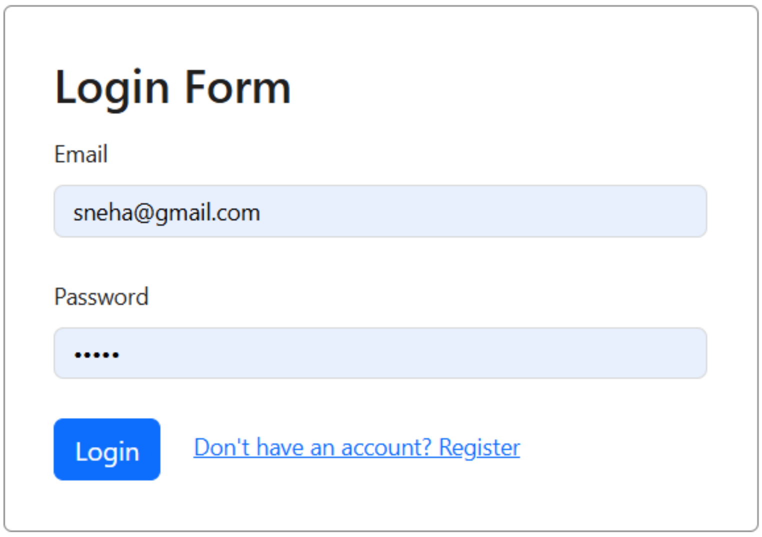
[3] R. Haux, "International Journal of Medical Informatics," *International Journal of Medical Informatics,* vol. 79, no. 9, pp. 599-610, September 2010.

[4] S. J. &. M. A. Vinod Kumar, "An Efficient Mutual Authentication Framework for Healthcare System in Cloud Computing," *Journal of Medical Systems,* vol. 142, no. 42, June 2018.

[5] J. H. D. T. A. Thierry Moulin, "Digital Health : The journal's pioneering journey 6 years on," *Digital Health,* vol. 205520762110340, no. 7, September 2021.

# APPENDICS

Login Page User



Login Form

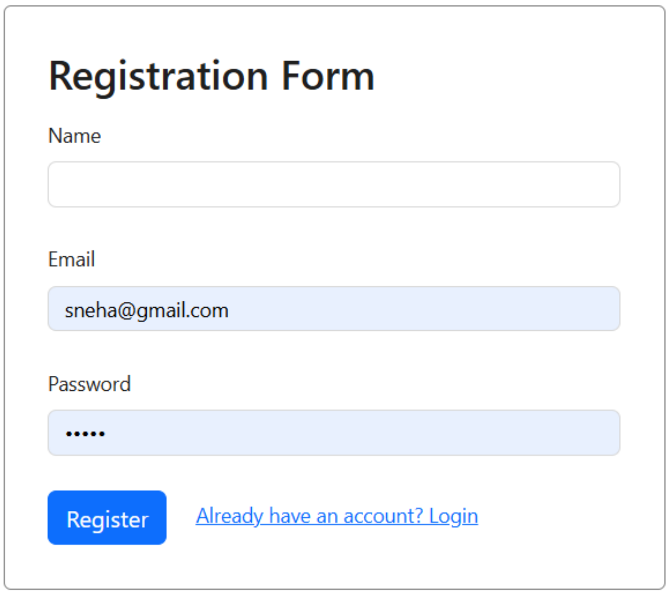
Email

sneha@gmail.com

Password

•••••

User Registration Page



Registration Form

Name

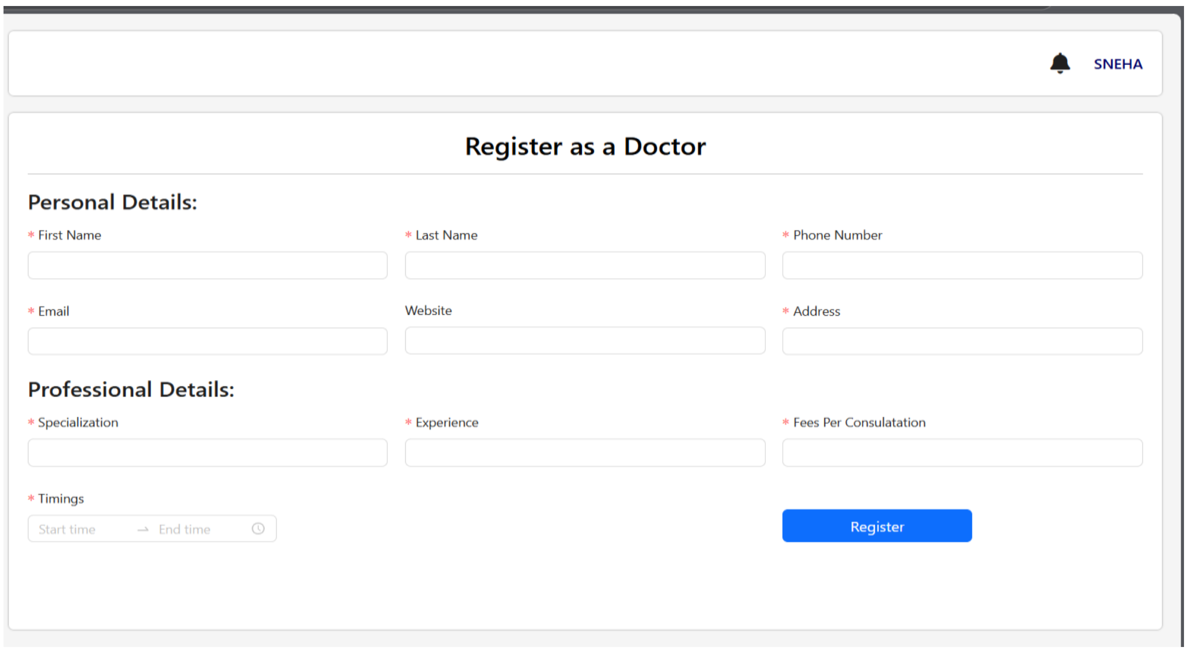
Email

sneha@gmail.com

Password

•••••

Doctor Registration Page



Register as a Doctor

Personal Details:

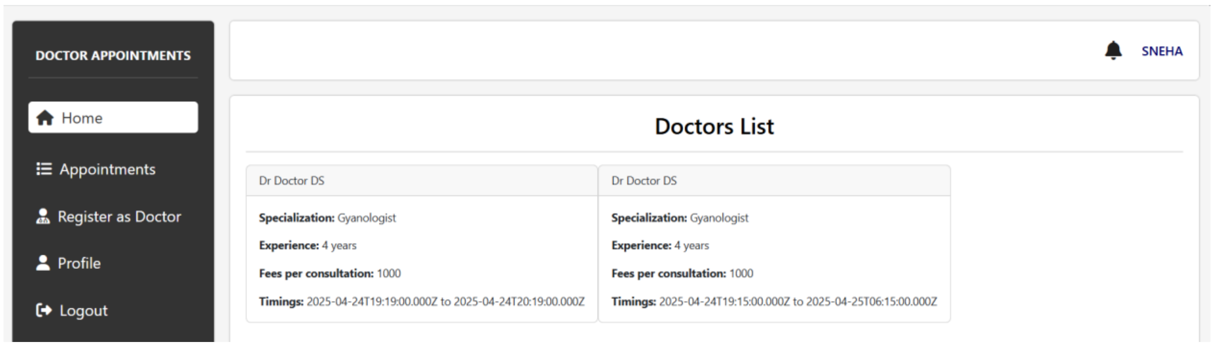
| | | |
|---|---|---|
| * First Name | * Last Name | * Phone Number |
| * Email | Website | * Address |

Professional Details:

| | | |
|---|---|---|
| * Specialization | * Experience | * Fees Per Consulatation |

* Timings

Home Page



Doctors List

Dr Doctor DS

**Specialization:** Gyanologist

**Experience:** 4 years

**Fees per consultation:** 1000

**Timings:** 2025-04-24T19:19:00.000Z to 2025-04-24T20:19:00.000Z

Dr Doctor DS

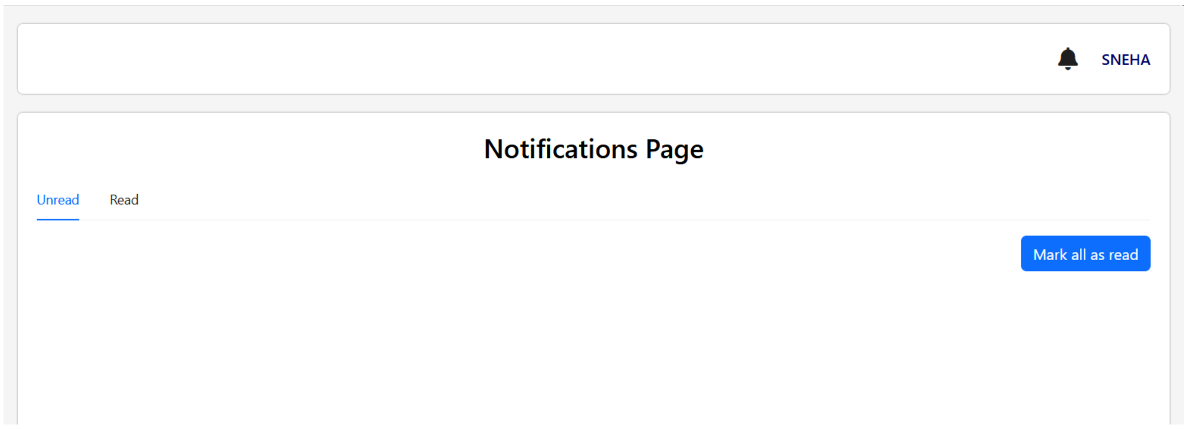
**Specialization:** Gyanologist

**Experience:** 4 years

**Fees per consultation:** 1000

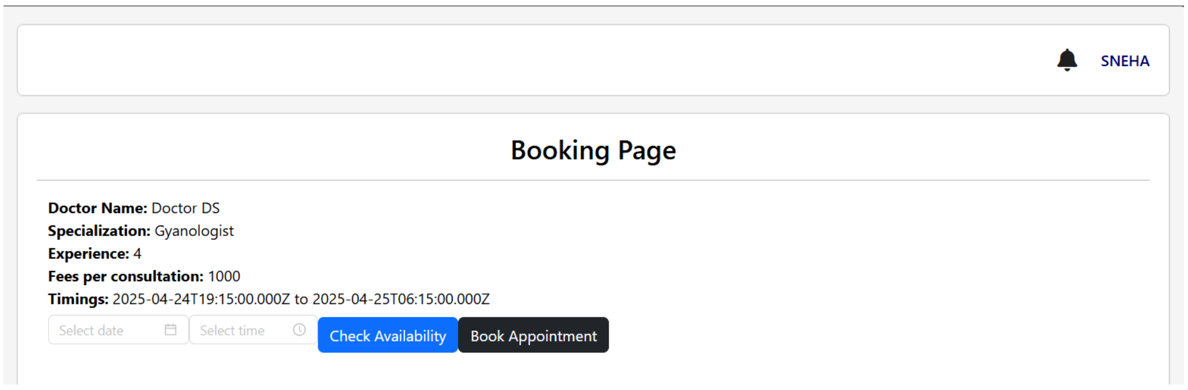
**Timings:** 2025-04-24T19:15:00.000Z to 2025-04-25T06:15:00.000Z

Notification Bar



Notifications Page

Booking Page

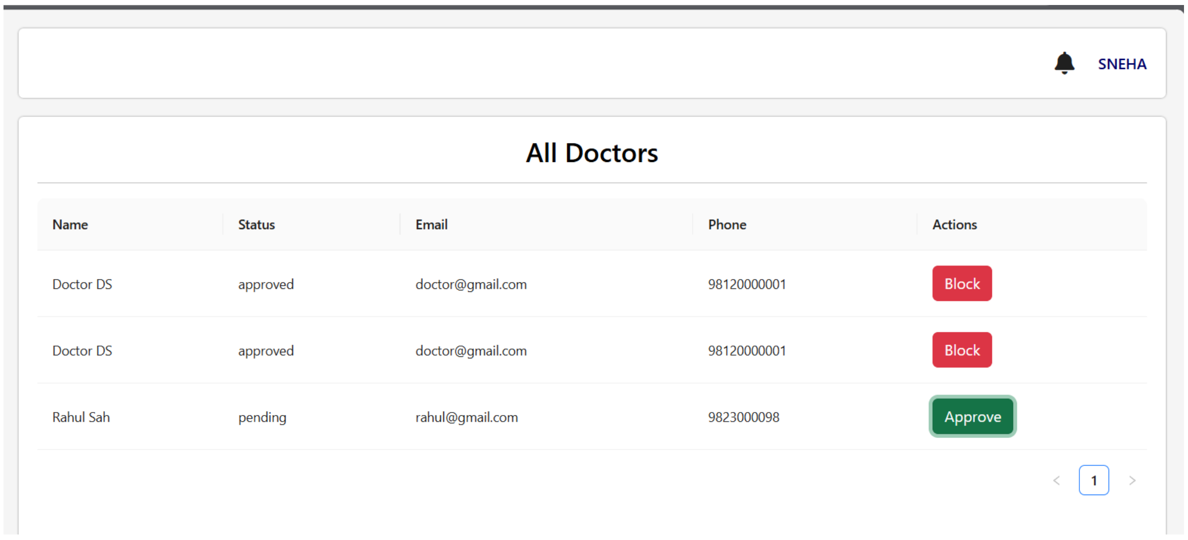


SNEHA

Booking Page

**Doctor Name:** Doctor DS
**Specialization:** Gyanologist
**Experience:** 4
**Fees per consultation:** 1000
**Timings:** 2025-04-24T19:15:00.000Z to 2025-04-25T06:15:00.000Z

Admin Page



SNEHA

All Doctors

| Name | Status | Email | Phone | Actions |
|---|---|---|---|---|
| Doctor DS | approved | doctor@gmail.com | 98120000001 | Block |
| Doctor DS | approved | doctor@gmail.com | 98120000001 | Block |
| Rahul Sah | pending | rahul@gmail.com | 9823000098 | Approve |