# HOSPITAL MANAGEMENT SYSTEM

**A Project Report**

Submitted in partial fulfillment of the
Requirements for the award of the Degree of

**BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)**

**By**

Ahmed Dilshad Shaikh
Seat Number: TY-323

**Under the esteemed guidance of**
**Prof. Mayur Tawade**

**Designation**

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**SHRI RAJASTHANI SEVA SANGH’S**
**SMT.PARMESHWARIDEVI DURGADUTT TIBERWALA LIONS**
**JUHU COLLEGE OF ARTS, COMMERCE & SCIENCE**

***(Affiliated to University of Mumbai)***

**JB NAGAR, ANDHERI EAST, MUMBAI MAHARASHTRA - 400059**

**2024 - 25**

SHRI RAJASTHANI SEVA SANGH'S
Smt. PARMESHWARIDEVI DURGADUTT TIBREWALA LIONS JUHU
COLLEGE OF ARTS, COMMERCE & SCIENCE
(AFFILIATED TO THE UNIVERSITY OF MUMBAI)

# CERTIFICATE

This is to certify that the project entitled, "**Hospital Management System**", is bonafied work of **Mr. Ahmed Dilshad Shaikh** bearing Seat.No: (**TY-323**) submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from University of Mumbai.

**Internal Guide** **Coordinator**

**External Examiner**

**Date:** **College Seal**

# PROFORMA FOR THE APPROVAL PROJECT PROPOSAL

**PNR No.: 2022016401545717** **Roll No.: 323**

1. Name of the Student

**MR. AHMED DILSHAD SHAIKH**

2. Title of the Project

**HOSPITAL MANAGEMENT SYSTEM**

3. Name of the Guide

**PROF. MAYUR TAWADE**

4. Teaching Experience of the Guide **2.5 Years**

5. Is this your first submission? Yes □ No □

Signature of the Student

Date: ………………

Signature of the Guide

Date: ………………

Signature of the Coordinator

Date: …………………

# ABSTRACT

The **Hospital Management System** (HMS) is designed to streamline and optimize hospital operations, offering a centralized platform for managing patient information, appointments, staff data, and medical records. It aims to automate key processes such as patient registration, appointment scheduling, reducing manual intervention and minimizing errors. By leveraging a combination of web technologies such as JSP, Servlets, and MySQL, the system provides secure access to data for both hospital staff and patients.

Our project Hospital Management System (HMS) includes registration of patients, storing their details into the system, and also booking their appointments with doctors. Our software has the facility to give a unique id for every patient and stores the details of every patient and the staff automatically. User can search availability of a doctor and the details of a doctor.

# ACKNOWLEDEGEMENT

I would like to express my deepest gratitude to all those who have supported and guided me throughout the development of my project, "**Hospital Management System**."

Firstly, I am profoundly grateful to my project advisor, **Prof. Mayur Tawade**, for their invaluable guidance, insightful feedback, and unwavering support. Their expertise and encouragement have been instrumental in the successful completion of this project.

I would also like to extend my thanks to **Mumbai University/Information Technology Department** for providing the resources and infrastructure necessary for the development of this project. The academic environment and facilities have greatly contributed to my learning experience.

Special thanks to my classmates and colleagues who have provided constructive feedback and assistance throughout the project. Their collaboration and camaraderie have been greatly appreciated.

Finally, I wish to acknowledge my family and friends for their understanding and encouragement during this period. Their patience and support have been a source of strength.

Thank you all for your contributions and support.

# DECLARATION

I hereby declare that the project entitled, "**Hospital Management System**" done at **SMT. PARMESHWARIDEVI DURGADUTT TIBREWALA LIONS JUHU COLLEGE OF ARTS COMMERCE & SCIENCE,** has not been in any case duplicated to submit to any other universities for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfilment of the requirements for the award of degree of **BACHELOR OF SCIENCE(INFORMATION TECHNOLOGY)** to be submitted as fifth semester project as part of our curriculum.

**Name and Signature of the Students**

# TABLE OF CONTENTS

# TABLE OF FIGURE

# CHAPTER - 1
# INTRODUCTION

In this busy world we don't have the time to wait in infamously long hospital queues. The problem is, queuing at hospital is often managed manually by administrative staff, then take a token there and then wait for our turn then ask for the doctor and the most frustrating thing - we went there by traveling a long distance and then we come to know the doctor is on leave or the doctor can't take appointments.

A **Hospital Management System** (HMS) will help us overcome all these problems because now patients can book their appointments at home, they can check whether the doctor they want to meet is available or not. Doctors can also confirm or decline appointments, this help both patient and the doctor because if the doctor decline appointment then patient will know this in advance and patient will visit hospital only when the doctor confirms the appointment this will save time and money of the patient.

## 1.1 Background

The healthcare industry requires seamless coordination and efficient management of information to ensure high-quality patient care. Hospital Management Systems (HMS) have emerged as vital tools to address these needs by providing comprehensive solutions for managing hospital operations.

An HMS facilitates the efficient management of patient information, appointments, medical records, and billing. It provides a centralized platform for storing and accessing data, enabling healthcare providers to retrieve patient details easily. Additionally, an HMS can significantly reduce the administrative burden on hospital staff by automating routine tasks such as appointment scheduling.

The Hospital Management System (HMS) developed in this project uses the Advance Java technology, it's include Servlet, JSP and JDBC/ MySQL for database connectivity to create an efficient, scalable, and user-friendly system.

### **Advantages of HMS over Traditional Systems**:

- **Appointment booking**: Helps patients cut the long queue and saves their time.
- **Overall cost reduction**: Cuts down paper cost as all the data are computerized.
- **Data security**: Helps to keep patients records private.

In conclusion, the implementation of a Hospital Management System is essential for modern healthcare institutions seeking to improve efficiency, reduce errors, and enhance patient care. By automating administrative tasks, ensuring regulatory compliance, and providing patients with better access to services, an HMS can transform the way hospitals operate.

## 1.2 Objectives

The key objectives of this project are designed to address the main pain points of manual hospital management systems. By automating core processes, the Hospital Management System (HMS) seeks to create a more efficient and user-friendly environment for both staff and patient.

1. **Streamline Hospital Operations:** Automate essential functions such as patient registration, appointment scheduling, and medical record management to reduce manual tasks and improve overall hospital efficiency.

2. **Enhance User Experience:** Offer patients a seamless experience with features like online appointment booking, access to medical records, and real-time notifications, improving patient satisfaction and reducing wait times.

3. **Reduce Operational Costs:** Lower cost associated with manual labor, paperwork, and inefficiencies, by automating hospital processes, thereby minimizing human errors and improving cost-effectiveness.

4. **Improve Healthcare Accessibility:** Increase access to healthcare services by enabling telemedicine, online consultations, and remote patient monitoring, extending the reach of the hospital’s services.

## 1.3 Purpose, Scope, and Applicability

### 1.3.1 Purpose

The primary purpose of this software will help the company to be more efficient in registration of their patients and manage appointments, records of patients. It enables doctors and admin to view and modify appointments schedules if required. The purpose of this project is to computerize all details regarding patient details and hospital details.

### 1.3.2 Scope

The Hospital Management System (HMS) will focus on the basic hospital management functions in its initial implementation. The scope of the system includes:

- **User Registration and Management**: New users, including patients and hospital staff (doctors, nurses, and administrators), will be able to sign up and manage their personal profiles. This feature includes basic information such as name, contact details, and role within the hospital.

- **Appointment Scheduling**: The system will allow patients to schedule appointments with doctors, and hospital staff can manage and update the schedule. Doctors will also be able to review their daily appointments.

- **Medical Record Management**: The system will manage the creation, updating, and storage of patients' medical records. This includes maintaining records of diagnoses, prescriptions, and past treatment history, ensuring easy access for healthcare providers.

- **Billing and Payments**: The system will generate and manage bills for hospital services provided to patients. It will allow hospital administrators to track payments and outstanding balances.

- **Inventory Management**: Hospital administrators will have full control over the inventory of medical supplies, equipment, and medications. The system will track usage, alert staff when stock is low, and help manage reorder processes.

In the future, the system can be extended to include more advanced features such as online transactions for pay their bills online through secure payment gateway and the integration of external health system. This flexibility ensures that the system will evolve to meet the growing and changing needs of the hospital over time.

### 1.3.3 Applicability

The Hospital Management System (HMS) has wide applicability across various types of healthcare facilities. It can be implemented in:

- **Public Hospitals**: Government-run hospitals can use the HMS to manage patient data, appointment scheduling, medical records, and billing more efficiently. This system can streamline operations, reduce paperwork, and improve patient care.
- **Private Hospitals**: Private healthcare institutions can use the HMS to offer enhanced services such as online appointment scheduling, medical record management, and billing. This system will allow private hospitals to focus on patient-centric care and reduce administrative burdens.
- **Clinics and Medical Centers**: Small to medium-sized clinics can benefit from the system by managing patient appointments, storing medical histories, and tracking inventory for medicines and medical supplies.
- **Specialized Health Facilities**: The HMS can be tailored for specific medical facilities like dental clinics, pediatric hospitals, psychiatric centers, and other specialized medical institutions. It can manage treatment histories, patient appointments, and inventory specific to their needs.
- **Telemedicine Services**: The system can be integrated with telemedicine platforms to provide remote consultations, manage virtual appointments, and allow patients to receive prescriptions digitally.

The flexibility and scalability of the system make it suitable for a wide range of healthcare environments, ensuring it can adapt to different institutional needs.

.

## 1.4 Achievements

Upon completing the development of the Hospital Management System (HMS**)**, the following key achievements were attained:

- **Proficiency in Full-Stack Development**: Working on the HMS project enhanced my skills in full-stack web development, using Java Servlets, JSP, and JDBC. I gained hands-on experience in developing and integrating both front-end and back-end components for a large-scale application.
- **Mastery of Relational Database Management**: Through the use of MySQL, I learned how to efficiently manage a relational database, including designing schemas, handling large datasets, and performing CRUD (Create, Read, Update, Delete) operations. I also became proficient in writing complex SQL queries to ensure optimized data retrieval and management.
- **Enhanced Understanding of Server-Side Programming**: By working extensively with Java Servlets, I gained in-depth knowledge of server-side programming, including managing HTTP requests and responses, implementing session management, and processing data on the server side.
- **Improved Front-End Development Skills**: Using JSP (JavaServer Pages), I improved my skills in creating dynamic web pages and managing user interactions. This included handling form data, displaying dynamic content, and creating an intuitive, user-friendly interface for hospital staff and patients.
- **Efficient Project Structuring and Deployment**: Throughout the project, I learned how to structure and organize a web-based application for maintainability and scalability. I also gained experience in deploying the application on Apache Tomcat and configuring the environment to ensure smooth operation.
- **Problem-Solving and Debugging Expertise**: The project presented challenges like optimizing database queries, handling patient data securely, and ensuring smooth interaction between the front-end and back-end. These challenges helped me develop strong problem-solving and debugging skills, ensuring efficient and reliable system performance.
- **Application of Security Best Practices**: Security was a critical component of the HMS. I learned to implement security measures such as data encryption, input validation, and role-based access control to prevent unauthorized access and ensure the protection of sensitive patient and hospital data.

## 1.5 Organization of Report

This report is structured into six chapters, each focusing on a different aspect of the project:

- **Chapter 1: Introduction** - Provides the background, objectives, and scope of the project.
- **Chapter 2: Review of Technologies** - Discusses the technologies used in the development of the project, including Java Servlet, JSP, JDBC/ MySQL and other relevant tools.
- **Chapter 3: Project Requirements and Analysis** - Outlines the system requirements and provides an in-depth analysis of the project scope, including functional and non-functional requirements.
- **Chapter 4: System Design** - Covers the system's design, including the database schema, user interface designs, and overall architecture.
- **Chapter 5: Implementation and Testing** - Presents the implementation process, coding details, and testing methodologies employed to ensure the system functions as intended.
- **Chapter 6: Conclusion and Future Enhancements** - Concludes the report by summarizing the outcomes and discussing potential future improvements.

# CHAPTER - 2
# SURVEY OF TECHNOLOGIES

The development of the Hospital Management System (HMS) required careful selection of technologies to ensure scalability, performance, and a seamless user experience. Each technology was chosen based on its suitability to meet the specific functional and non-functional requirements of the system. Below is a survey of the technologies explored and chosen for developing the HMS:

1. **Java Servlet and JSP: Server-Side Logic and Dynamic Web Pages**

    - **Java Servlets**: Servlets were chosen as the core technology for handling server-side logic. They process requests from the client, interact with the database, and generate dynamic content for the users. Servlets offer a reliable, scalable way to develop web-based applications that require server-side computation.

    - **JSP (JavaServer Pages)**: JSP was used to create dynamic web pages that integrate seamlessly with the back-end logic. JSP allows the embedding of Java code in HTML, simplifying the process of presenting dynamic content such as patient records, appointment schedules, and billing information.

2. **JDBC: Database Connectivity**

    - **JDBC (Java Database Connectivity)** was selected as the primary tool for interacting with the MySQL database. It allows for efficient database connectivity and operations, including executing SQL queries, performing CRUD operations, and handling transactions. JDBC provides a flexible and standardized way to connect Java applications with databases.

3. **Bootstrap: Responsive and User-Friendly Design**

    - **Responsive Design**: Automatically adapts to different devices, ensuring that the HMS works well on desktops, tablets, and mobile devices.

    - **Pre-built Components**: Includes elements such as buttons, forms, tables, alerts, and navigation menus, allowing for faster UI development with a consistent, modern look.

    - **Customizable Themes**: Easily customize the visual appearance of the HMS using Bootstrap’s utility classes and built-in themes.

# CHAPTER - 3
# REQUIREMENTS AND ANALYSIS

## 3.1 Problem Definition

In traditional hospital management systems, operations such as managing patient records, scheduling appointments, and handling billing processes are often carried out manually or with outdated software. These methods lead to various issues, such as inefficiencies, errors, and challenges in managing large volumes of patient and hospital data. The key problems to address are:

- **Inefficiency**: Manual processes like patient registration, appointment scheduling, and billing are time-consuming and error-prone, leading to delays and inaccuracies in hospital operations.
- **Limited Access**: Traditional systems provide little or no online access, restricting patients and staff from easily managing appointments, viewing medical records, or making payments remotely.
- **Data Management**: Handling large amounts of patient data manually increases the risk of data loss, inconsistencies, and makes generating reports and managing medical history challenging.

The Hospital Management System (HMS) aims to solve these problems by digitizing hospital operations, improving accessibility for patients and staff, and optimizing data management processes.

## 3.2 Requirements Specification

### 3.2.1 Functional Requirements

Functional requirements specify the features and functionalities the MS must provide to meet user needs:

- **User Registration and Authentication:** The system should allow patients, doctors, and staff to register and securely log in. Each type of user should have role-based access to the system's functionalities.

- **Patient Management:** Hospital staff should be able to add, update, and manage patient information, including personal details, medical history, and ongoing treatment plans.

- **Appointment Scheduling:** Patients should be able to book appointments with doctors, and doctors should be able to manage and view their schedules. The system must prevent double bookings and provide notifications for upcoming appointments.

### 3.2.2 Non-Functional Requirements

Non-functional requirements define the system's quality attributes:

- **Performance:** The system should have fast response times and handle concurrent user interactions efficiently.
- **Security:** User data and hospital information must be protected against unauthorized access and breaches.
- **Usability:** The interface should be intuitive and easy to navigate for users of all technical levels.
- **Reliability:** The system should be robust and available with minimal downtime, ensuring consistent operation.

## 3.3 Planning and Scheduling

Planning and scheduling involve outlining the development phases, milestones, and timeline for the HMS project:

### 3.3.1 Development Phases

- **Phase 1:** Requirements Gathering and Analysis
  - **Activities:** Define problem, gather requirements, and document specifications.
- **Phase 2:** System Design
  - **Activities:** Design system architecture, database schema, and user interface.
- **Phase 3:** Implementation
  - **Activities:** Develop front-end and back-end components, integrate systems.
- **Phase 4:** Testing
  - **Activities:** Conduct unit testing, integration testing, and user acceptance testing.
- **Phase 5:** Deployment and Training
  - **Activities:** Deploy the system, provide user training, and documentation.
- **Phase 6:** Maintenance and Support
  - **Activities:** Provide support, update the system, and handle user feedback.

### 3.3.2 Milestones

- Completion of Requirements Specification
- Approval of System Design
- Completion of Core Development
- Successful Testing and Bug Fixing
- System Deployment and User Training

## 3.4 Software and Hardware Requirements

### 3.4.1 Software Requirements

**Server-Side**

- Operating System: Ubuntu Server or Windows Server
- Java Development Kit (JDK): Java JDK 8 or higher
- Apache Tomcat: Web server for hosting Java Servlets and JSP
- JDBC: Java Database Connectivity for interacting with the MySQL database

**Client-Side**

- HTML/CSS: For structuring and styling the user interface
- Bootstrap: For responsive and modern UI components

**Development Tools**

- Code Editor/IDE: Eclipse, IntelliJ IDEA, or NetBeans for Java development
- Version Control: Git and GitHub
- Database Management Tool: MySQL Workbench for managing and designing the database schema

### 3.4.2 Hardware Requirements

**Server**

- **Processor:** Multi-core CPU
- **Memory:** 8 GB RAM
- **Storage:** 100 GB SSD
- **Network:** High-speed internet connection

**Client Devices**

- **Processor:** Modern multi-core CPU
- **Memory:** 4 GB RAM
- **Browser:** Latest versions of Chrome, Firefox, Edge, or Safari

**Core Features:**

- **Staff Management:** Registration, login, and profile management
- **Patient Management:** Patient records, Appointment Scheduling
- **Billing System:** Invoicing, Payment Processing
- **Reporting:** Data Analytics, Compliance Reporting

## 3.5 Preliminary Product Description

The Hospital Management System (HMS) is designed to modernize the traditional methods of managing healthcare operations, ensuring efficiency, accuracy, and accessibility. The primary goal of the system is to automate and streamline core hospital functions, such as managing patient data, tracking appointments, processing billing, and enabling online access for both patients and healthcare providers.

### 3.5.1 Objectives of the System:

- **Automate Healthcare Operations**: Reduce manual errors and inefficiencies by digitizing core processes like patient registration, appointment scheduling, and billing management.
- **Enhance User Experience**: Provide an intuitive interface that allows patients and healthcare providers to easily navigate the system, manage appointments, and access patient records
- **Improve Accessibility**: Enable online access to the hospital system, allowing users to schedule appointments and view medical records.
- **Efficient Data Management**: Utilize a centralized database to manage patient information, appointment schedules, billing records, and medical histories, ensuring data consistency, security, and scalability.

### 3.5.2 Functions of the System:

- **User Management**: The system allows for user registration and login with appropriate authentication to ensure secure access. Users (patients and healthcare providers) will have the ability to update their profiles, view their medical history, and manage appointments.
- **Patient Records Management**: The HMS provides a comprehensive database where healthcare providers can access and update patient records, including medical history, medications, allergies, and treatment plans.
- **Appointment Scheduling**: Users can schedule, reschedule, and cancel appointments through the system. The system tracks appointment availability in real-time and sends notifications for upcoming appointments.
- **Administrative Features**: Administrators and healthcare staff will have access to advanced functionalities such as managing user accounts, overseeing appointment schedules, and generating reports on hospital operations.

### 3.5.3 System Operation:

The Hospital Management System (HMS) operates as a web-based platform, accessible via a browser from any device with an internet connection. The system utilizes Java Servlets and JavaServer Pages (JSP) for dynamic content generation and server-side processing. The back-end is designed to manage business logic and database interactions through JDBC to ensure seamless communication with a MySQL database..

By implementing these functionalities, the Hospital Management System enhances the quality of healthcare services, streamlines administrative tasks, and addresses the inefficiencies commonly found in traditional healthcare management.

## 3.6 Conceptual Model

The conceptual model illustrates the high-level design and interaction between different components of the HMS:

### 3.6.1 System Architecture

- **Front-End:** Developed using JavaServer Pages (JSP), providing a user-friendly and interactive interface for both patients and healthcare providers.
- **Back-End:** Built with Java Servlets to manage server-side logic, processing user requests, and handling business rules.
- **Database:** Utilizes MySQL for structured and reliable data storage, ensuring data integrity and efficient query processing.

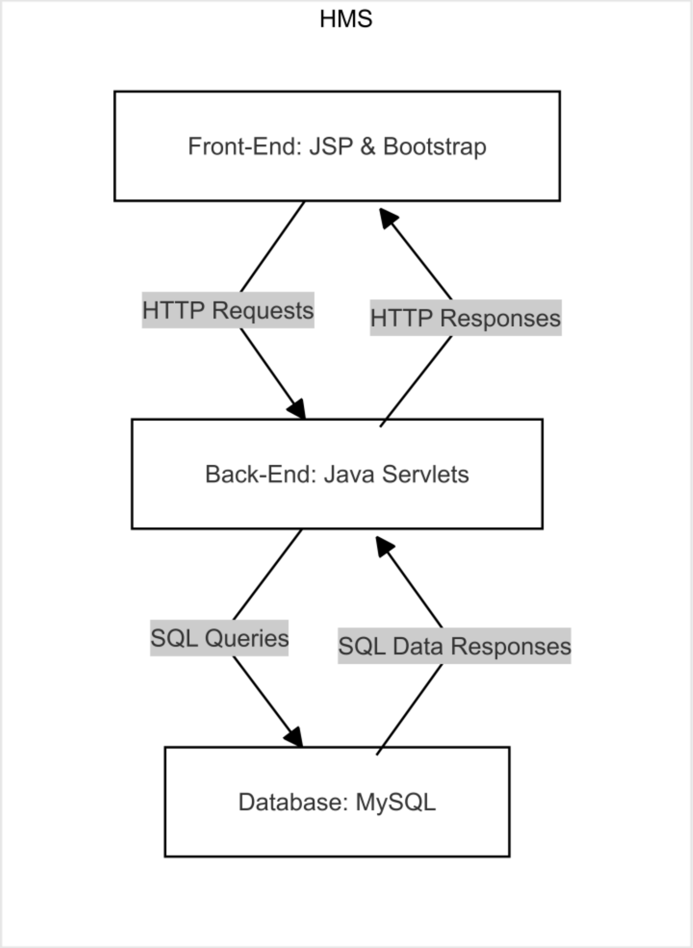


Fig-3.1 System Architecture

### 3.6.2 User Interaction Flow

- **User Registration/Login:** Users interact with the registration or login page, which communicates with the back-end for authentication.
- **Patient Management:** Healthcare providers can access the patient management interface to view and update patient records.
- **Appointment Scheduling:** Users can browse available appointment slots and schedule appointments with healthcare providers.
- **Billing and Payment:** Patients can view their bills and make payments through the system.

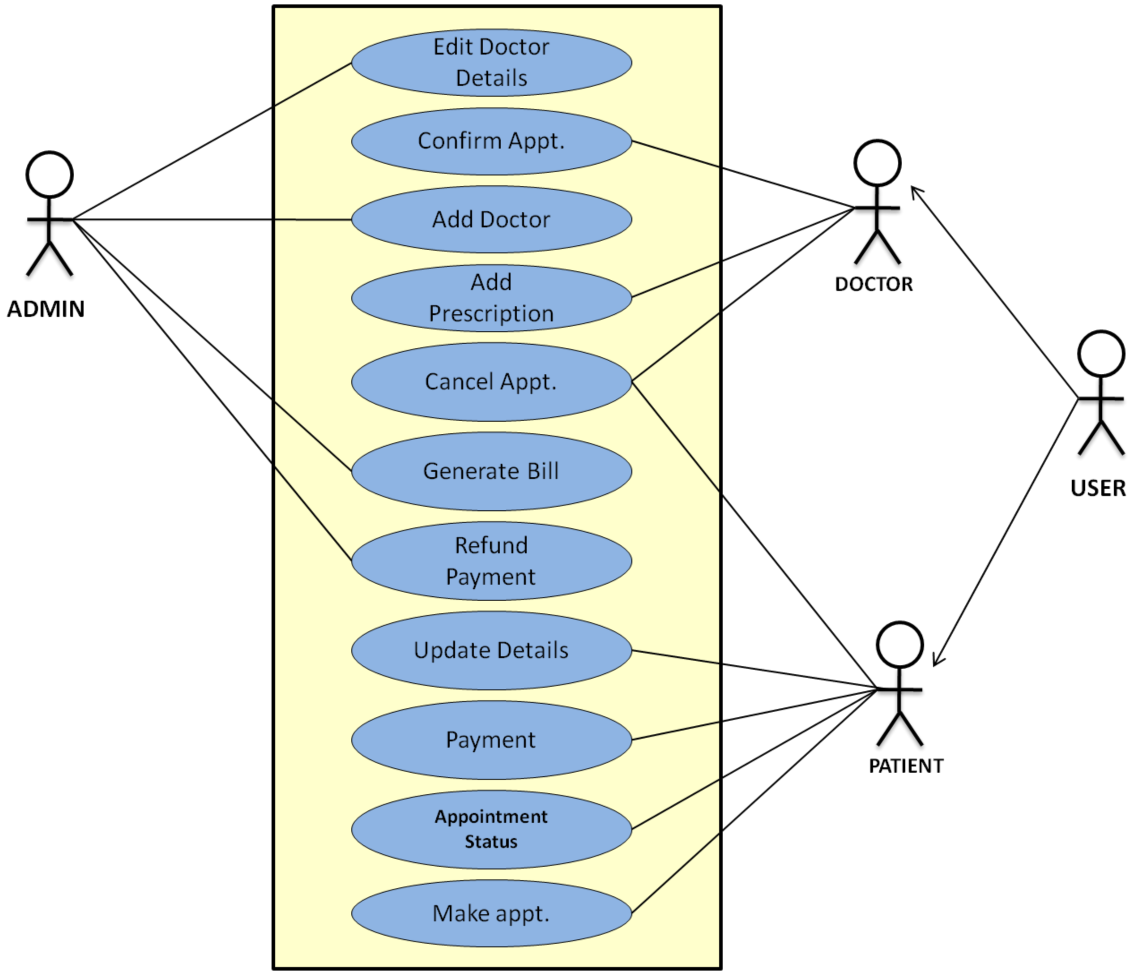


Fig-3.2 User Interaction Flow

## 3.6.3 Data Flow

- **Client Requests:** User actions on the front-end generate requests that are sent to the server via specific endpoints defined in the Java Servlets.
- **Server Processing:** The server receives these requests, processes the business logic associated with them, interacts with the MySQL database to retrieve or modify data, and prepares responses.
- **Database Operations:** Data is stored, retrieved, and updated in the MySQL database based on the operations requested by the users.

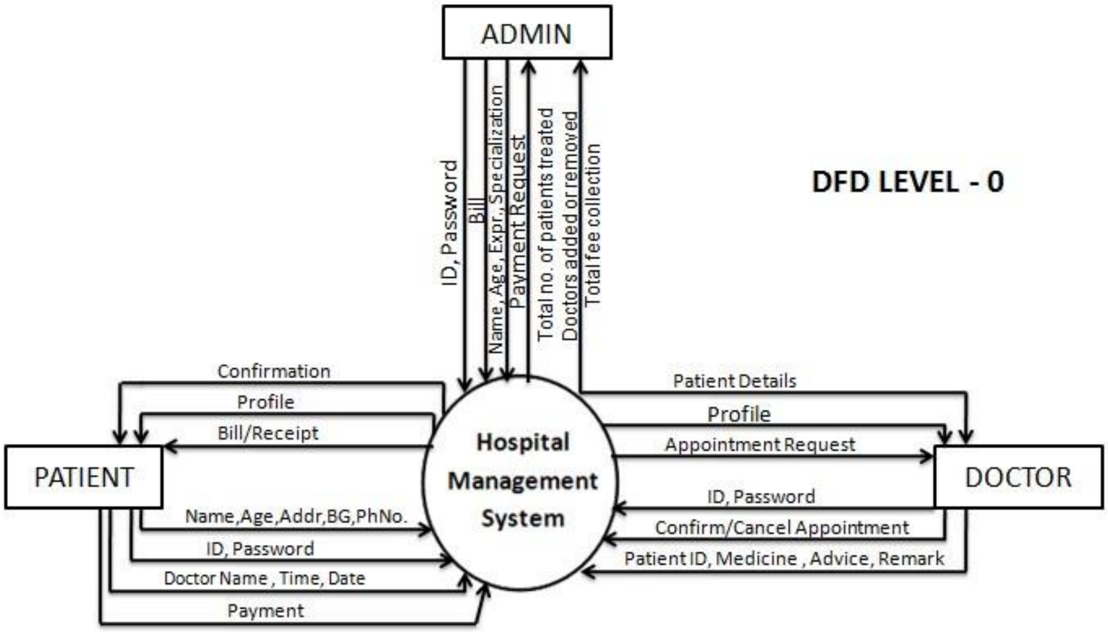


Fig-3.3 Data Flow Diagram

# CHAPTER - 4
# SYSTEM DESIGN

## 4.1 Basic Modules

The system is divided into the following basic modules:

1. **Admin Module:** Provides the admin with the ability to manage doctors, patients, view appointments and handle other administrative tasks such as payment.

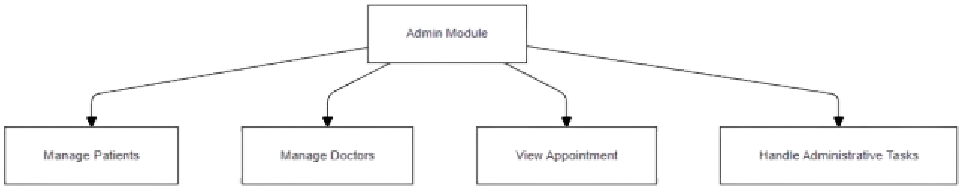


Fig-4.1 Admin Module

2. **Patient Module:** Handles user registration, login, viewing doctors, booking appointments, and managing profiles.

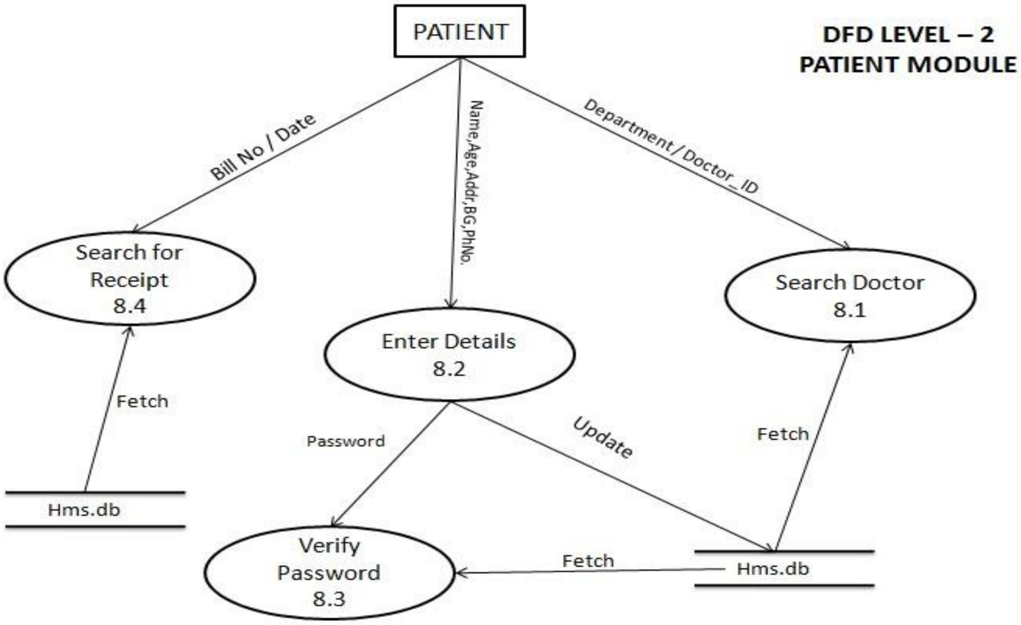


Fig-4.2 Patient Module

3. **Doctor Module:** Doctors can log in, view appointments, update patient records, and set their availability.

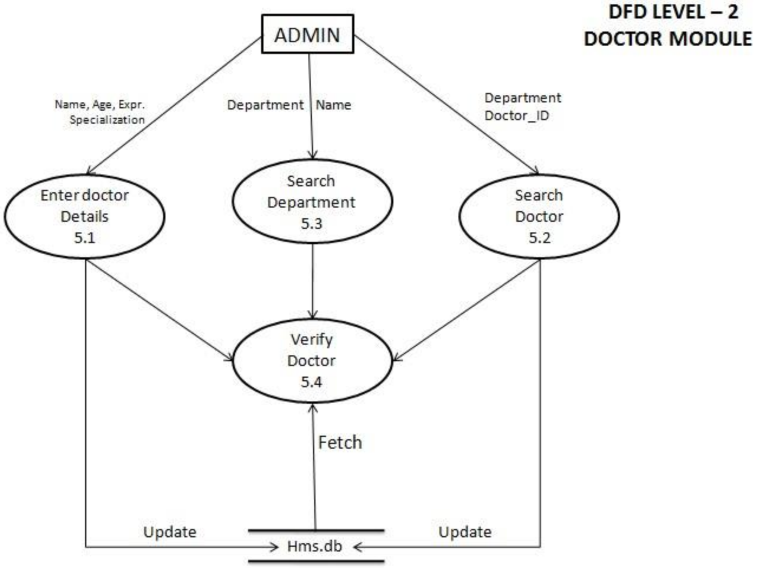


Fig-4.3 Doctor Module

4. **Appointment Management Module:** Manages booking, canceling, rescheduling, and tracking appointments.

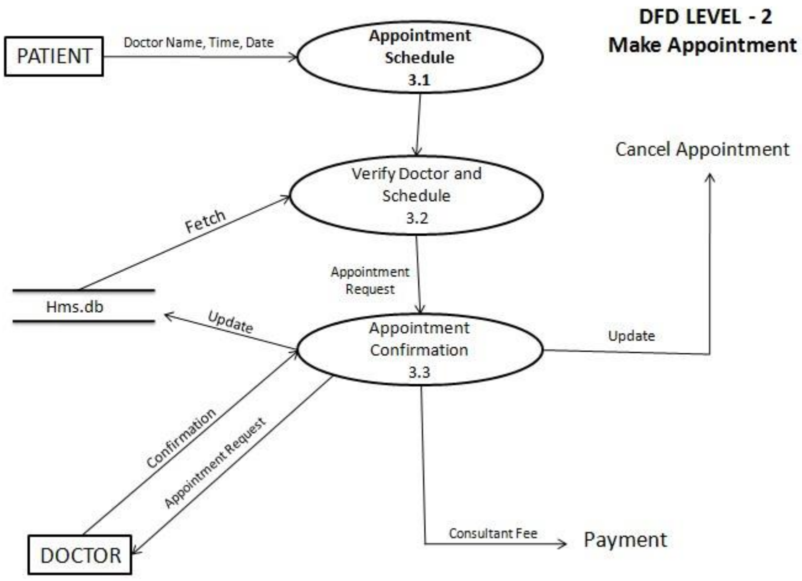


Fig-4.4 Appointment Module

5. **Billing and Payment Module:** Handles bill generation, payments, and payment history.

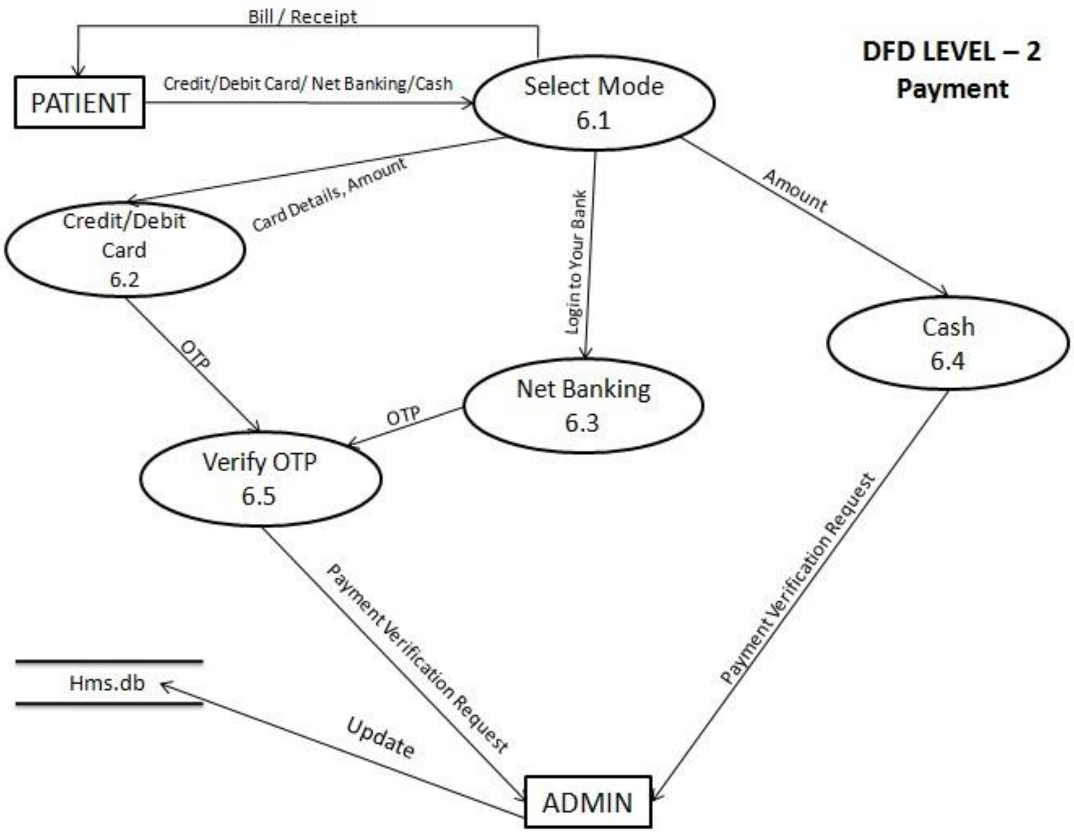
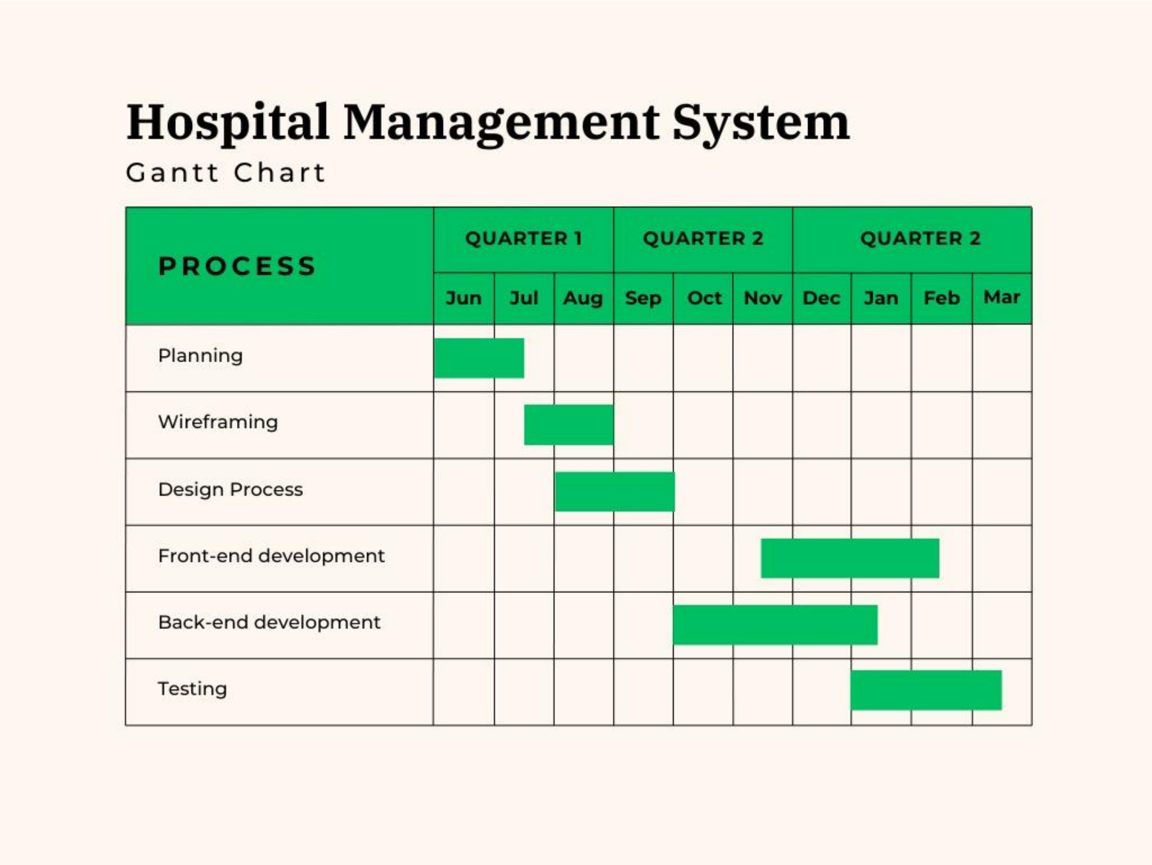


Fig-4.5 Payment Module

- **Project Timeline:** The following Gantt chart shows the development timeline for each of the basic modules mentioned above, including their design, implementation, and testing phases.



Hospital Management System

Gantt Chart

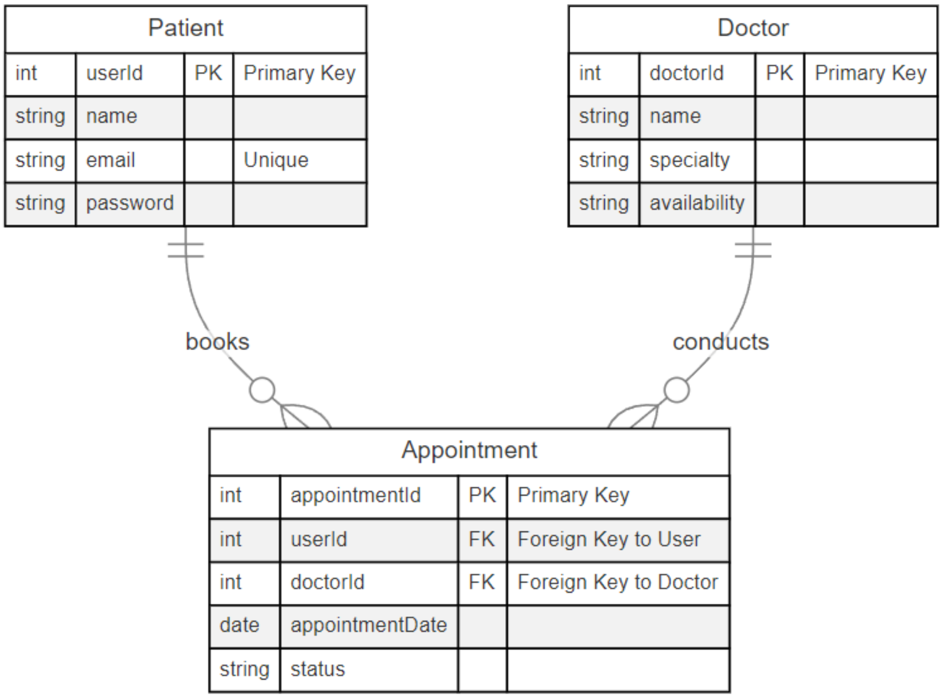
| PROCESS | QUARTER 1 | | | QUARTER 2 | | | QUARTER 2 | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Jun | Jul | Aug | Sep | Oct | Nov | Dec | Jan | Feb | Mar |
| Planning | 🟩 | 🟩 | | | | | | | | |
| Wireframing | | 🟩 | 🟩 | | | | | | | |
| Design Process | | | 🟩 | 🟩 | | | | | | |
| Front-end development | | | | | | 🟩 | 🟩 | 🟩 | 🟩 | |
| Back-end development | | | | | 🟩 | 🟩 | 🟩 | 🟩 | | |
| Testing | | | | | | | | 🟩 | 🟩 | 🟩 |

Fig- 4.6 Gantt Chart Design

## 4.2 Data Design

### 4.2.1 Schema Design

The database design includes several entities the primary entities include Admin, Patients, Doctors and Appointments. The relationships between these entities are shown in the ER diagram.



| Patient | | | |
|---|---|---|---|
| int | userId | PK | Primary Key |
| string | name | | |
| string | email | | Unique |
| string | password | | |

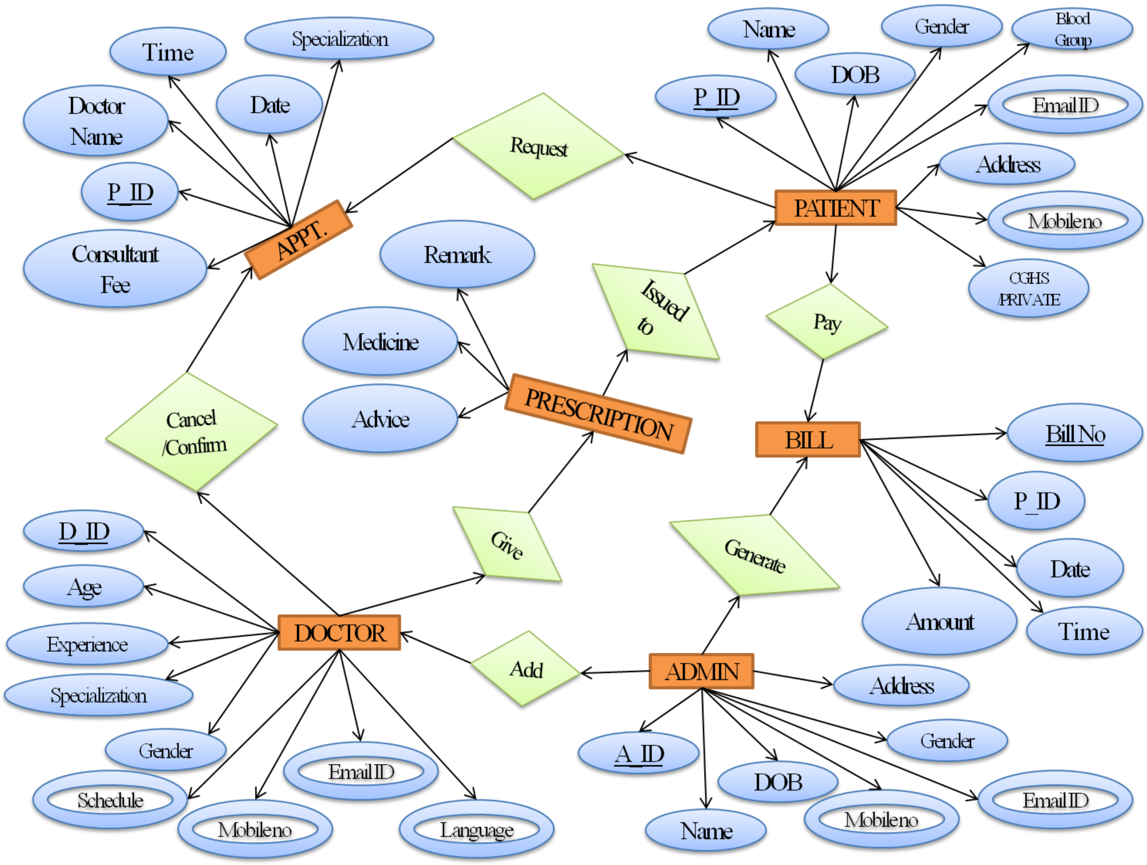
| Doctor | | | |
|---|---|---|---|
| int | doctorId | PK | Primary Key |
| string | name | | |
| string | specialty | | |
| string | availability | | |

| Appointment | | | |
|---|---|---|---|
| int | appointmentId | PK | Primary Key |
| int | userId | FK | Foreign Key to User |
| int | doctorId | FK | Foreign Key to Doctor |
| date | appointmentDate | | |
| string | status | | |

Fig-4.6 Schema Design(ERD)

## ✧ ER Model

The ER Model outlines the entities, attributes, and relationships within the Hospital Management System:



### 4.2.2 Data Integrity and Constraints

To ensure data integrity within the Hospital Management System, several constraints are implemented across different tables. The following table outlines the entities, their attributes, and the constraints applied to maintain data integrity

| S NO. | COLUMN NAME | DATA TYPE | CONSTRAINTS | DESCRIPTION |
|---|---|---|---|---|
| 1. | P_ID | Integer | Primary Key | Contains Unique Id |
| 2. | Name | Varchar(50) | - | Contains Name |
| 3. | DOB | Varchar(50) | - | Contains Date Of Birth |
| 4. | Gender | Varchar(50) | - | Contains Gender |
| 5. | Blood Group | Varchar(50) | - | Contains Blood Group |
| 6. | Email ID | Varchar(50) | Unique, Not Null | Contains Email Id |
| 7. | Address | Varchar(50) | - | Contains Address |
| 8. | Mobile No. | Integer | Unique, Not Null | Contains Mobile No. |

Table: Patient

| S NO. | COLUMN NAME | DATA TYPE | CONSTRAINTS | DESCRIPTION |
|---|---|---|---|---|
| 1. | P_ID | Integer | Primary Key | Contains Unique Id Patient |
| 2. | Specialization | Varchar(50) | - | Contains Name of the Department in which Patient wants to visit |
| 3. | Doctor’s Name | Varchar(50) | - | Contains Doctor Name Patient Wants To Visit |
| 4. | Date | Date | - | Contains Date For The Appointment |

Table: Appointment

| S NO. | COLUMN NAME | DATA TYPE | CONSTRAINTS | DESCRIPTION |
|---|---|---|---|---|
| 1. | D_ID | Integer | Primary Key | Contains unique ID |
| 2. | Age | Integer | - | Contains age |
| 3. | Gender | Varchar(50) | - | Contains  gender |
| 4. | Specialization | Varchar(50) | - | Contains specialization |
| 7. | Mobile No. | Integer | Unique, Not Null | Contains mobile number |
| 8. | Email ID | Varchar(50) | Unique, Not Null | Contains Email Id |
| 9. | Schedule | Varchar(50) | - | Contains day and time for which the doctor is available |

Table: Doctor

| S NO. | COLUMN NAME | DATA TYPE | CONSTRAINTS | DESCRIPTION |
|---|---|---|---|---|
| 1. | A_ID | Integer | Primary Key | Contains unique ID. |
| 2. | Name | Varchar(50) | - | Contains Name |
| 5. | Email ID | Varchar(50) | - | Contains Email Id |
| 6. | Password | Integer | - | Contains Password |

Table: Admin

Fig-4.9 Data Integrity & Constraints(Table)

# 4.3 Procedural Design

## 4.3.1 Logic Diagrams

Logic diagrams include flowcharts and activity diagrams to represent the process flow of key functions such as user login, search doctor, and appointment process. For example, the flow of a appointment will involve checking the availability of the doctor, registering the users, and updating the appointment status.

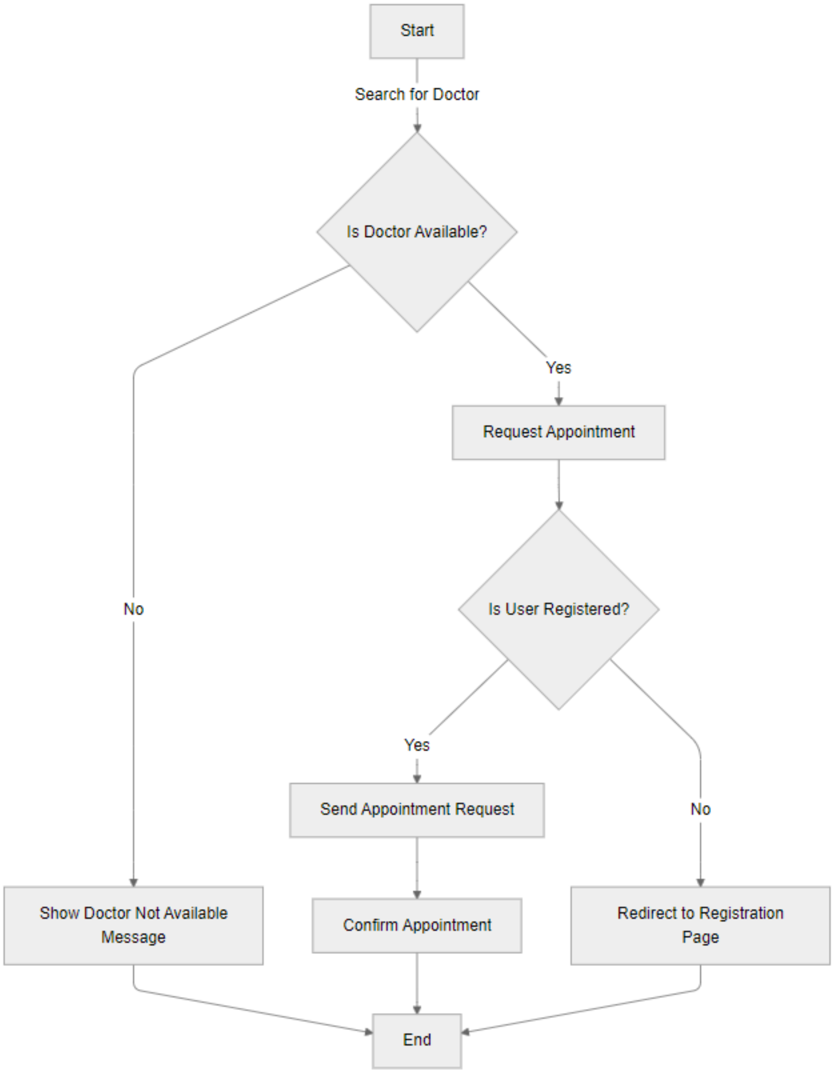


Fig-4.8 Logic Diagram

### 4.3.2 Data Structures

The data structures used in the system are optimized for managing a large collection of patients, doctors, and medical records. Hash maps and arrays are used for efficient storage and retrieval of data, ensuring that search and filtering operations on patient records, doctor schedules, and appointment bookings can be performed quickly.

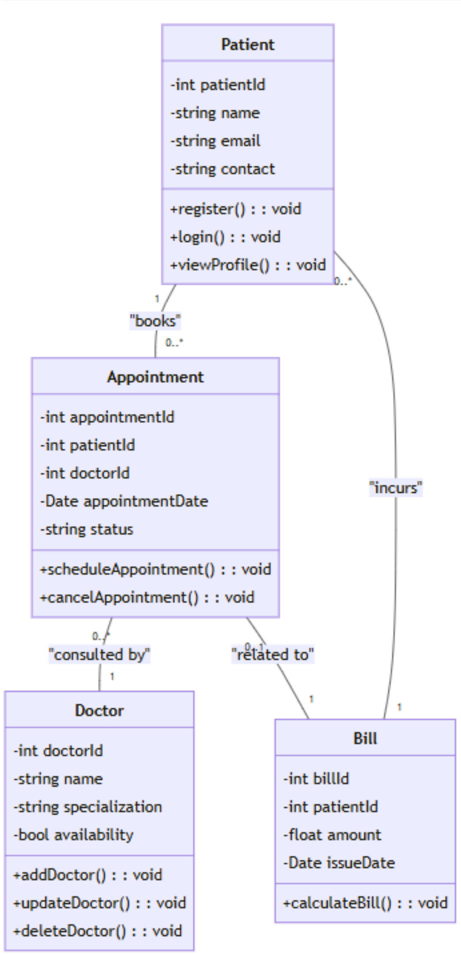


Fig-4.9 Class Diagram

### 4.3.3 Algorithms Design

The algorithms implemented in the system handle operations such as searching for doctors, and managing appointment requests. The doctor search algorithm is optimized to filter based on name. The appointment algorithm is based on the doctor approval.

## 4.4 User Interface Design

The user interface is designed to be intuitive and user-friendly. Wireframes are created to demonstrate the layout of the login page, dashboard, book appointment, and admin management panel. Responsive design principles are applied to ensure compatibility across different devices and screen sizes.

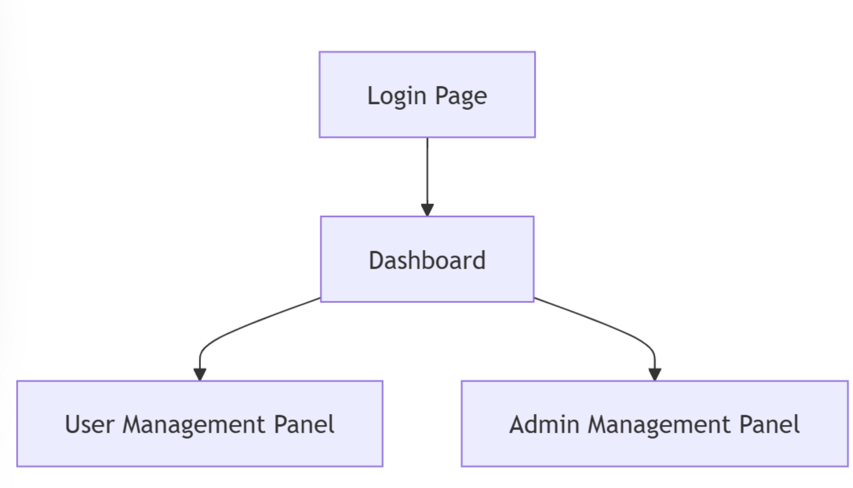


Fig-4.10 UI Design

## 4.5 Security Issues

The system incorporates several security measures to protect user data and prevent unauthorized access. The system utilizes session-based authentication to ensure that only authenticated users can access restricted areas. Role-based access control (RBAC) is implemented to differentiate between admin and user functionalities. diagrams demonstrate how different roles interact with the system for secure operations.

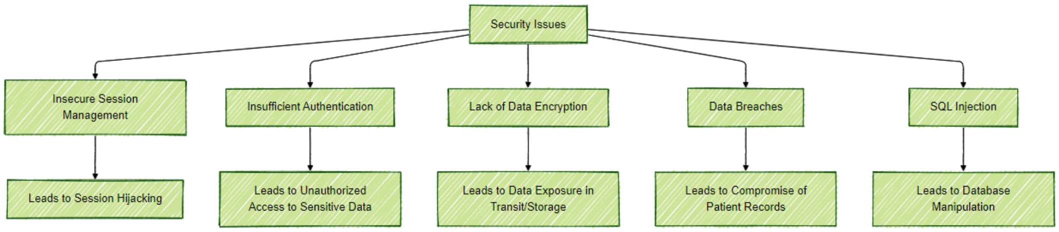

Security Issues
Insecure Session Management
Insufficient Authentication
Lack of Data Encryption
Data Breaches
SQL Injection
Leads to Session Hijacking
Leads to Unauthorized Access to Sensitive Data
Leads to Data Exposure in Transit/Storage
Leads to Compromise of Patient Records
Leads to Database Manipulation

Fig-4.11 Security Diagram

## 4.6 Test Cases Design

Test cases are designed to verify the functionality and performance of each module. Unit testing is conducted to ensure individual components are working as expected, and integration testing verifies that the modules work together seamlessly. Sample test cases include testing the user registration process, verifying doctor functionality, and checking the appointment algorithm.

| **Test Case ID** | **Test Case Description** | **Preconditions** | **Test Steps** | **Expected Result** | **Actual Result** | **Pass/Fail** |
|---|---|---|---|---|---|---|
| TC_01 | Verify patient registration functionality | User is on the patient registration page | 1. Open registration page<br>2. Enter valid details (name, email, password)<br>3. Click "Submit" | New patient is successfully registered and displayed in the patient list (data saved to MySQL) | Patient was successfully registered | Pass |
| TC_02 | Validate error message for missing required fields during patient registration | User is on the patient registration page | 1. Open the registration page<br>2. Leave "name" or "age" fields empty<br>3. Click "Submit" | System displays "All fields are required" error message | Error message displayed as expected | Pass |
| TC_03 | Verify doctor appointment booking functionality | User (patient) is logged in and on the appointment booking page | 1. Select a valid doctor from the dropdown<br>2. Choose a date and time<br>3. Click "Book Appointment" | Appointment is successfully booked and confirmation message is displayed (appointment details saved to MySQL) | Appointment was booked successfully | Pass |

| | | | | | | |
|---|---|---|---|---|---|---|
| TC_04 | Validate error message for booking an appointment outside working hours | User is logged in and on the appointment booking page | 1. Select a valid doctor<br>2. Choose a time outside the doctor’s available hours<br>3. Click "Book Appointment" | System displays an error message "Appointment time is outside working hours | Error message displayed as expected | Pass |
| TC_05 | Verify error handling for duplicate patient registration | User is on the patient registration page | 1. Enter a patient ID that already exists in the system<br>2. Click "Submit" | System displays "Patient with this ID already exists" error message | Error message displayed as expected | Pass |
| TC_06 | Verify bill calculation for a patient after discharge | User (admin or accountant) is on the billing page | 1. Open the billing section for a discharged patient<br>2. Ensure services and medication costs are added correctly<br>3. Click "Generate Bill" | Bill is generated correctly with accurate charges (data fetched from MySQL) | Bill was generated correctly | Pass |

Fig-4.12 Test Case Diagram