A Project Report
HOSPITAL MANAGEMENT SYSTEM

Project submitted to the

SRM University – AP, Andhra Pradesh

for the partial fulfillment of the requirements to award the degree of

Bachelor of Technology

In

Computer Science and Engineering

School of Engineering and Sciences

Submitted by

Shaik Yaseen Shannu AP2211001118
Tadepalli Naga Vaishnavi AP22110011220
Kudumala Devi Priya AP22110011221
Gummadapu Sai Geetha AP22110010764
Paturu Vijaya Lakshmi AP22110011392

Under the Guidance of

P N V Syamala Rao M

Assistant professor, Dept.ofCSE

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh – 522 240

April-2025

CERTIFICATE

This is to certify that the project work titled **" Hospital Management System using Java"** has been successfully carried out under our supervision. The project embodies genuine and original work conducted by the students and adheres to the standards required for submission to **SRM University–AP** for the award of the **Bachelor of Technology** degree under the **School of Engineering and Sciences.**

This work reflects the students' independent efforts, critical thinking, and technical understanding of the subject area. The study and implementation of the **Hospital Management System using Java** have been completed in accordance with academic guidelines and demonstrate a solid grasp of the topic.

We find the work to be suitable for academic recognition and in fulfillment of the requirements of the **B.Tech program.**

Tadepalli Naga Vaishnavi – (AP22110011220)
Shaik Yaseen Shannu – (AP22110011128)
Gummadapu Sai Geetha – (AP22110010764)
Kudumala Devi Priya – (AP22110011221)
Paturu Vijaya Lakshmi – (AP22110011392)

Supervisor
P N V Syamala Rao M

ACKNOWLDEGMENT

We express our sincere gratitude to our mentor, **P N V Syamala Rao M**, for his valuable guidance, encouragement, and expertise throughout the duration of this project. His support and constructive feedback have been instrumental in shaping our understanding and ensuring the successful completion of this work.

We extend our appreciation to SRM University–AP for providing us with the necessary resources and a conducive environment for academic learning and innovation.

We also acknowledge the contributions of our team members:

Gummadapu Sai Geetha – (AP22110010764)
(Database Design & GUI Implementation)
Handled the creation of the **MySQL database** and designed **core parts of the GUI**, ensuring smooth **connectivity** and user interaction across all modules.

Shaik Yaseen Shannu – (AP22110011128)
(Patient Management – Add & Update)
Implemented the functionalities to **add new patients** and **update patient records**, ensuring accurate and up-to-date information handling.

Paturu Vijaya Lakshmi – (AP22110011392)
(Patient Deletion & Doctor Management)
Took charge of **adding doctor profiles** and implemented the feature to **delete patient records**, maintaining database integrity and modularity.

Kudumala Devi Priya – (AP22110011221)
(Doctor Update & Deletion)
Worked on **updating doctor details** and enabling **doctor deletion**, allowing the admin to manage doctor information dynamically.

Tadepalli Naga Vaishnavi – (AP22110011220)
(Appointment Module)
Developed the logic and interface for **booking and canceling appointments**, allowing efficient scheduling and doctor-patient coordination.

This collaborative effort brought together **database management, GUI design, and core functional modules** to create a complete and efficient Hospital Management System. We combined technical expertise with a focus on usability to deliver a reliable solution for hospital workflows.

ABSTRACT

The Hospital Management System is a desktop-based application developed using Java Swing for the graphical user interface (GUI) and MySQL for backend data management. This system is designed to automate and streamline essential hospital operations, enabling efficient management of patient and doctor records, as well as appointment scheduling. It offers key features, including patient management, doctor management, and appointment booking and cancellation, ensuring smooth coordination between patients, doctors, and administrative staff for effective healthcare delivery.

The system emphasizes accurate data handling, security, and user-friendly interactions. It leverages Java's object-oriented programming structure, which ensures a modular, maintainable codebase, and MySQL's relational database capabilities for robust data storage and retrieval. This combination provides a seamless integration between the frontend and backend, ensuring the system is both responsive and reliable.

Key functionalities of the system include the ability to add, update, and delete patient and doctor records, along with the ability to book and cancel appointments based on real-time doctor availability. The system is designed to reduce manual errors, improve operational efficiency, and ensure accurate management of hospital data.

By simulating real-world hospital workflows, this project offers valuable insights into healthcare software development and follows best practices in modular design, error handling, and user input validation. It demonstrates the application of Java and MySQL in developing real-time, enterprise-level solutions, providing a strong foundation for further healthcare system development

TABLE OF CONTENTS

INTRODUCTION

In today's fast-paced world, hospitals need efficient systems to manage their operations smoothly. The **Hospital Management System** is a desktop application designed to help hospitals manage patient and doctor information, appointments, and other essential tasks easily. The system allows hospital staff to register patients, manage doctor details, and schedule or cancel appointments through a simple, user-friendly interface.

Built using **Java Swing** for the interface and **MySQL** for storing data, the system ensures secure and quick handling of information. It helps reduce paperwork and the chances of errors while improving overall hospital operations.

This project is a great tool for students and developers interested in building software for the healthcare industry. It shows how **Java** and **MySQL** can be used to create real-world applications that improve hospital management. By automating routine tasks, the system helps hospitals save time, reduce mistakes, and provide better care to patients.

The **Hospital Management System** demonstrates how technology can make hospital operations more efficient and improve the experience for both hospital staff and patients.

METHODOLOGY

The Hospital Management System was developed using a structured and step-by-step approach to ensure smooth functionality and a user-friendly interface. The development was divided into several phases to ensure each feature was tested and implemented effectively. Below are the main stages followed during the development:

1. Requirement Analysis
The key features of the system were identified: **patient management**, **doctor management**, and **appointment scheduling**. We decided to use **Java Swing** for the GUI and **MySQL** for storing and managing hospital data.

2. Design Phase
The system layout was created using **JFrames** and **JPanels**, focusing on simplicity and ease of use. **CardLayout** was utilized to manage the flow between different sections of the system, such as patient registration and appointment booking.

3. Implementation Phase
Each feature was developed as a separate module using **Object-Oriented Programming (OOP)** principles. Data storage and retrieval were handled by **MySQL**, while input validation and error handling ensured smooth user interactions.

4. Testing Phase
Each module was tested thoroughly with various inputs to ensure proper functionality. For example, we tested the patient and doctor record management and appointment booking to ensure no errors occurred during use.

5. Final Integration
All modules were integrated into a cohesive system, ensuring smooth data flow and interaction between features. The entire system was tested end-to-end to ensure that all components worked together seamlessly.

OBJECTIVES

1. To Create a Realistic Hospital Management Environment

The project aims to build a desktop application that simulates real-world hospital operations. It enables users to manage patient records, doctor profiles, and appointments, offering a seamless experience similar to actual hospital management systems.

2. To Provide a Secure and Reliable Data Management Platform

Security is crucial in managing sensitive healthcare information. This system ensures that patient and doctor records are securely stored and accessed only by authorized personnel, minimizing data breaches and errors during transactions such as appointment bookings.

3. To Implement Object-Oriented Programming in Java

The project utilizes Object-Oriented Programming (OOP) concepts such as classes, objects, inheritance, and encapsulation to organize and structure the code efficiently. These principles ensure code reusability, easy maintenance, and future updates.

4. To Use Event-Driven Programming Techniques

User interactions, such as adding new patient records, booking appointments, and navigating through the system, are handled using Java's event-driven model. This approach ensures smooth, responsive, and dynamic user experiences.

5. To Design an Easy-to-Use GUI with Java Swing

A key objective is to provide a simple, user-friendly interface using Java Swing components like JButton, JTextField, JLabel, and JComboBox. The layout is designed to ensure ease of navigation and efficient task completion for hospital staff.

6. To Store and Access Hospital Data Efficiently

Patient, doctor, and appointment data are stored in a MySQL database, ensuring quick retrieval, secure updates, and efficient data management. This allows the system to handle large datasets effectively and securely.

7. To Build a Scalable and Maintainable Codebase

The system is developed in a modular way, with each feature (e.g., patient management, doctor profiles, appointment scheduling) handled in separate classes. This modular approach simplifies code maintenance and enables easy updates or additions, such as integrating more advanced features or transitioning to a web-based platform.

TECHNOLOGIES USED

Programming Language: Java

- Java was chosen as the core programming language for developing the entire Hospital Management System.
- It supports Object-Oriented Programming (OOP) principles, offering advantages such as modularity, reusability, and easy maintenance of the code.
- Java's platform independence ensures that the application can run on different operating systems without modification, making it suitable for diverse environments.

GUI Framework: Java Swing

- Java Swing was used to develop the graphical user interface (GUI) for the system.
- Swing provides a wide range of GUI components, such as JFrame, JPanel, JLabel, JButton, JTextField, and JComboBox, which are essential for creating an interactive and user-friendly interface.
- CardLayout was used to switch between different screens, such as patient management, doctor profiles, and appointment scheduling, ensuring a smooth flow between the modules.

Database Management: MySQL

- MySQL was used to handle the storage and retrieval of hospital data, such as patient records, doctor profiles, and appointment details.
- It provides a reliable relational database management system (RDBMS) for ensuring data integrity, security, and quick access to large datasets.

Event Handling: ActionListener Interface

- ActionListener was used to manage user interactions within the GUI, such as clicking buttons to add, update, or delete patient and doctor records, or to book and cancel appointments.
- This event-driven approach enables the system to respond dynamically to user input and perform the corresponding actions on the backend.

Error Handling: Try-Catch Blocks

- To ensure the system remains stable, try-catch blocks were used throughout the application to handle potential exceptions, such as database connection failures or invalid user inputs.
- This approach prevents the application from crashing and provides helpful feedback to users.

Data Persistence: MySQL Database

- Data, including patient, doctor, and appointment details, is stored persistently in the MySQL database. This ensures data is not lost when the system is closed and allows for efficient updates and retrieval of information as needed.

MODULES IMPLEMENTED

1. Patient Management Module

- Allows adding new patients with details like name, age, and gender
- Enables updating existing patient records
- Provides functionality to delete patients and their associated appointments
- Input validation for age and required fields

2. Doctor Management Module

- Supports adding new doctors with name and specialization
- Allows updating doctor information
- Provides functionality to delete doctors and their associated appointments
- Input validation for required fields

3. Appointment Scheduling Module

- Enables booking appointments between patients and doctors
- Validates date format and ensures appointments aren't scheduled in the past
- Checks doctor availability for the selected date
- Displays appointments in a tabular format

4. Appointment Cancellation Module

- Allows cancellation of existing appointments by ID
- Verifies appointment existence before cancellation

5. Data Display Module

- Shows all patients, doctors, and appointments in separate tables
- Refreshes data automatically after each operation
- Maintains ID-name mappings for dropdown selections

6. User Interface Module

- Provides tabbed interface for easy navigation between functions
- Includes form validation and error handling
- Features confirmation dialogs for delete operations
- Maintains consistent styling across all components

7. Database Connectivity Module

- Manages all database operations using JDBC
- Handles SQL queries and transactions
- Maintains referential integrity when deleting records

8. Combo Box Population Module

- Dynamically updates patient and doctor dropdowns
- Formats display as "ID - Name" for user-friendly selection

USER INTERFACE DESIGN

Main Components

- JFrame serves as the main application window
- JTabbedPane organizes functionality into three main tabs: Patients, Doctors, and Appointments
- JTable components display data in tabular format with scrollable views
- JPanel containers organize related UI elements

Input Components

- JTextField for text/numeric input (patient/doctor names, ages, IDs)
- JComboBox for gender selection and choosing patients/doctors in appointments
- JButton for all actions (add, update, delete, book appointments)

Data Display

- DefaultTableModel powers the JTables for displaying patient, doctor, and appointment data
- JScrollPane ensures tables are scrollable when data exceeds visible area

Security Features

1. Input Validation

- All fields are validated before database operations
- Age validation ensures reasonable values (1-120)
- Date validation for appointments (YYYY-MM-DD format)
- Empty field checks for required information

2. Data Integrity

- Foreign key constraints maintained (appointments reference valid patients/doctors)
- Cascading deletes handled explicitly (deleting patients/doctors also removes their appointments)
- Confirmation dialogs for destructive operations

3. Database Security

- Prepared statements used throughout to prevent SQL injection
- Sensitive database credentials stored as constants (should be externalized in production)

System Flow

1. Application Launch
- Loads MySQL JDBC driver
- Initializes main window with three tabs
- Populates initial data from database

2. Patient Management
- Add: Enter name, age, gender → validate → insert to database
- Update: Select patient by ID → modify fields → validate → update database
- Delete: Enter patient ID → confirm → delete from database (with related appointments)

3. Doctor Management
- Add: Enter name, specialization → validate → insert to database
- Update: Select doctor by ID → modify fields → validate → update database
- Delete: Enter doctor ID → confirm → delete from database (with related appointments)

4. Appointment Management
- Book: Select patient and doctor from dropdowns → enter date → validate → check availability → book
- Cancel: Enter appointment ID → confirm → remove from database

5. Data Refresh
- All tables automatically refresh after modifications
- Dropdown selections update when patient/doctor data changes

6. Application Exit

- Clean shutdown via Exit button

Implementation Details

Database Interaction

- Uses JDBC with MySQL
- Connection details configured as constants
- All operations use prepared statements

Error Handling

- Comprehensive try-catch blocks
- User-friendly error messages
- Stack traces logged for debugging

Validation Logic

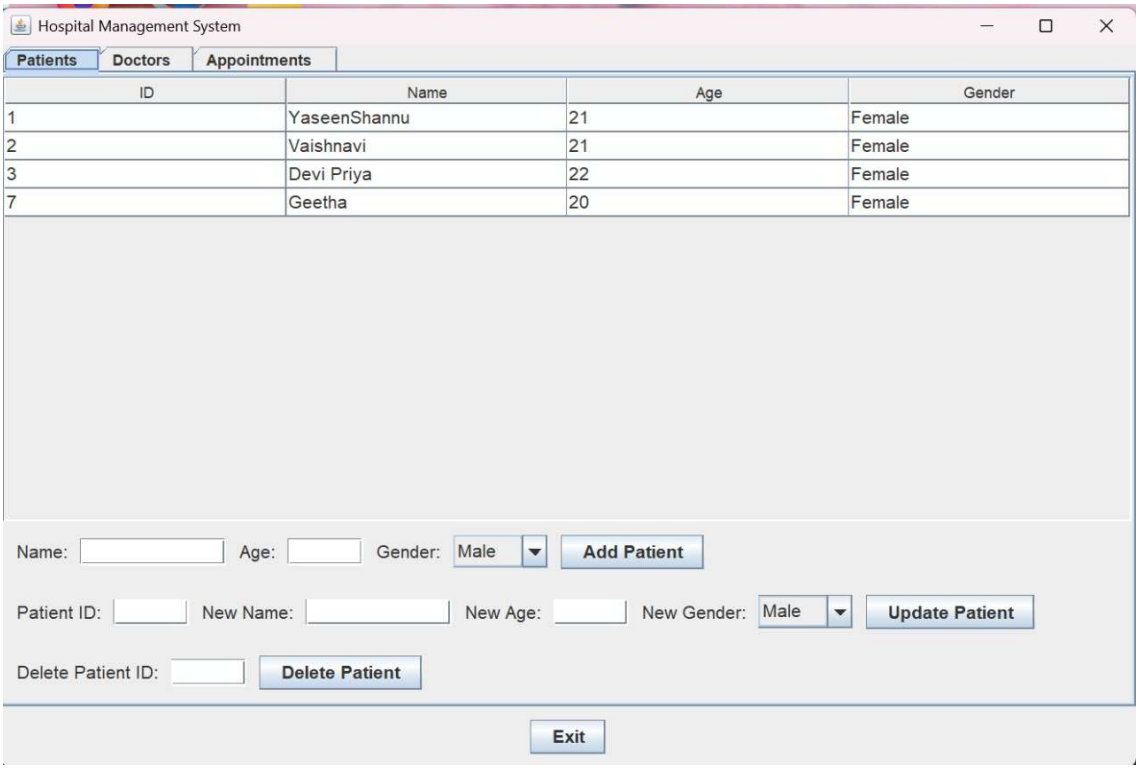
- Age must be positive integer (1-120)
- Dates must be valid and not in past
- All required fields must be populated
- Doctor availability checked before booking

RESULTS

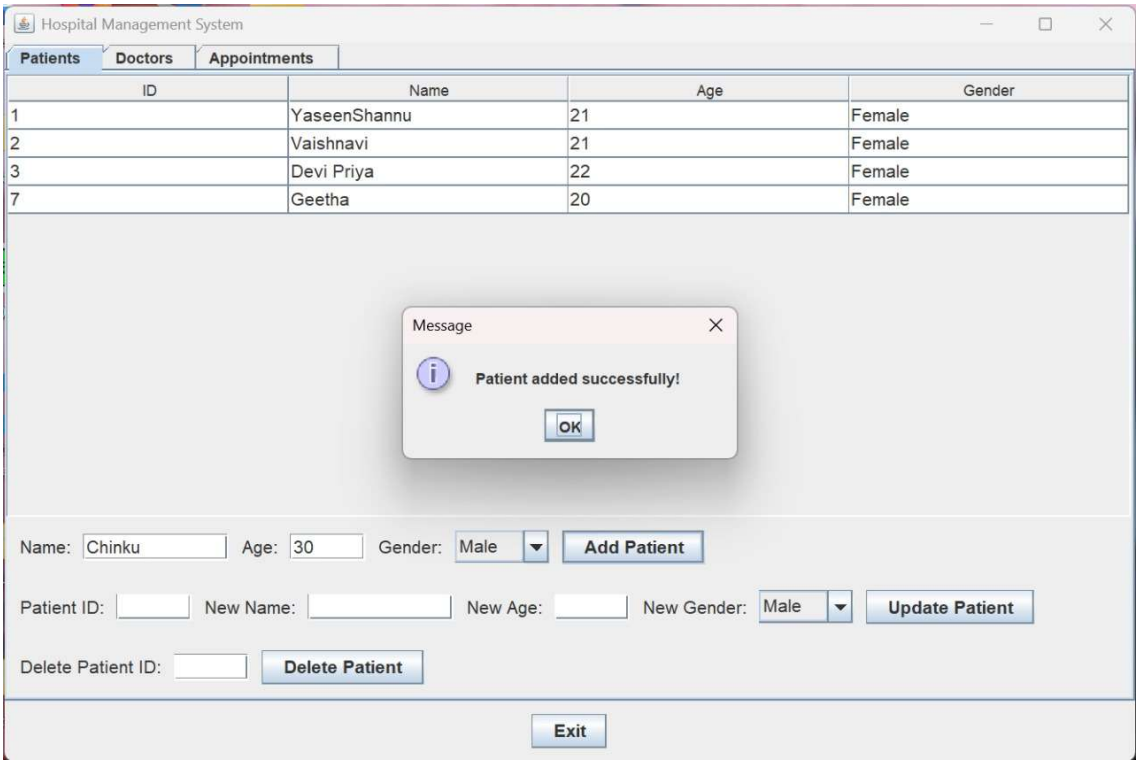
Database

```
mysql> use hospital;
Database changed
mysql> describe patients;
+--------+--------------+------+-----+---------+----------------+
| Field  | Type         | Null | Key | Default | Extra          |
+--------+--------------+------+-----+---------+----------------+
| id     | int          | NO   | PRI | NULL    | auto_increment |
| name   | varchar(255) | NO   |     | NULL    |                |
| age    | int          | NO   |     | NULL    |                |
| gender | varchar(10)  | NO   |     | NULL    |                |
+--------+--------------+------+-----+---------+----------------+
4 rows in set (0.06 sec)

mysql> describe doctors;
+----------------+--------------+------+-----+---------+----------------+
| Field          | Type         | Null | Key | Default | Extra          |
+----------------+--------------+------+-----+---------+----------------+
| id             | int          | NO   | PRI | NULL    | auto_increment |
| name           | varchar(255) | NO   |     | NULL    |                |
| specialization | varchar(255) | NO   |     | NULL    |                |
+----------------+--------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)

mysql> describe appointments;
+------------------+------+------+-----+---------+----------------+
| Field            | Type | Null | Key | Default | Extra          |
+------------------+------+------+-----+---------+----------------+
| id               | int  | NO   | PRI | NULL    | auto_increment |
| patient_id       | int  | NO   | MUL | NULL    |                |
| doctor_id        | int  | NO   | MUL | NULL    |                |
| appointment_date | date | NO   |     | NULL    |                |
+------------------+------+------+-----+---------+----------------+
```

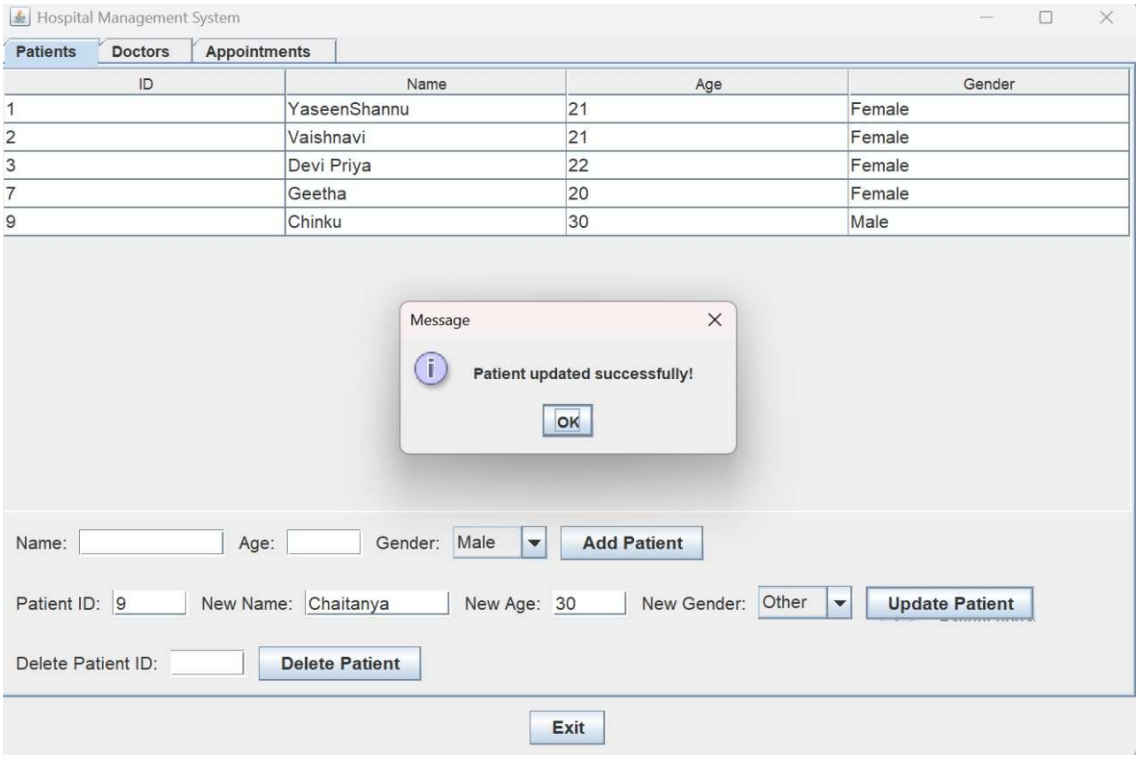
View Patients



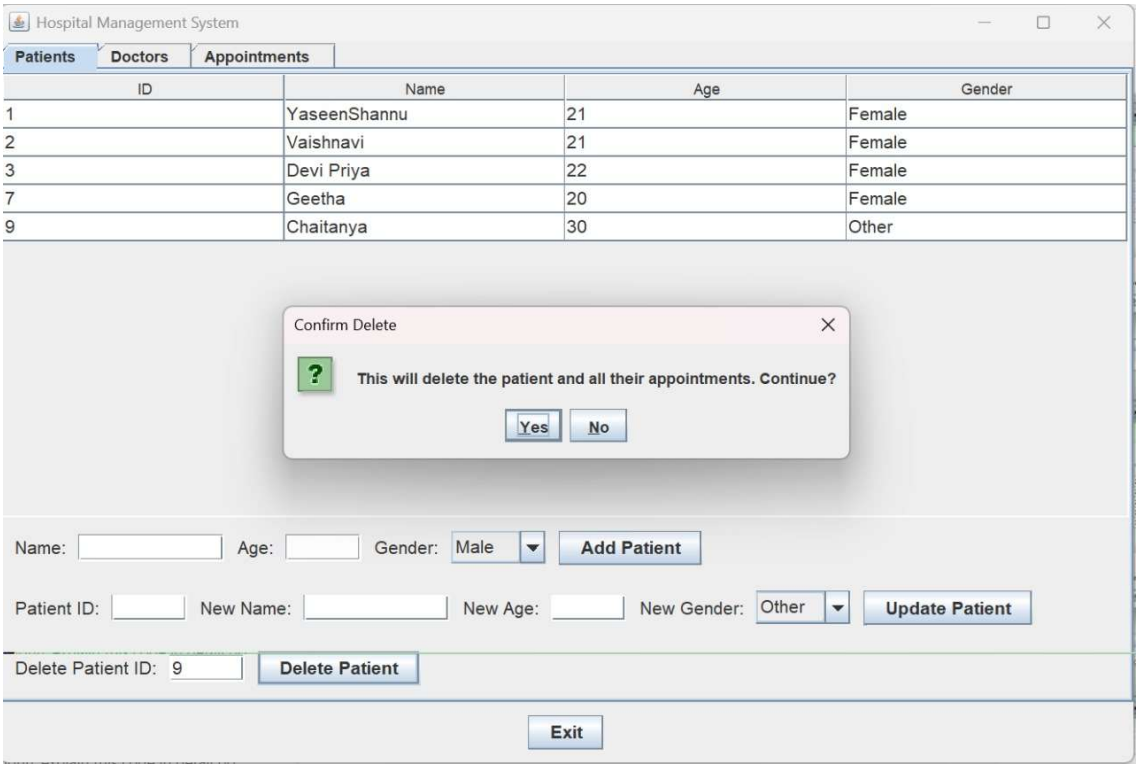
Add Patient



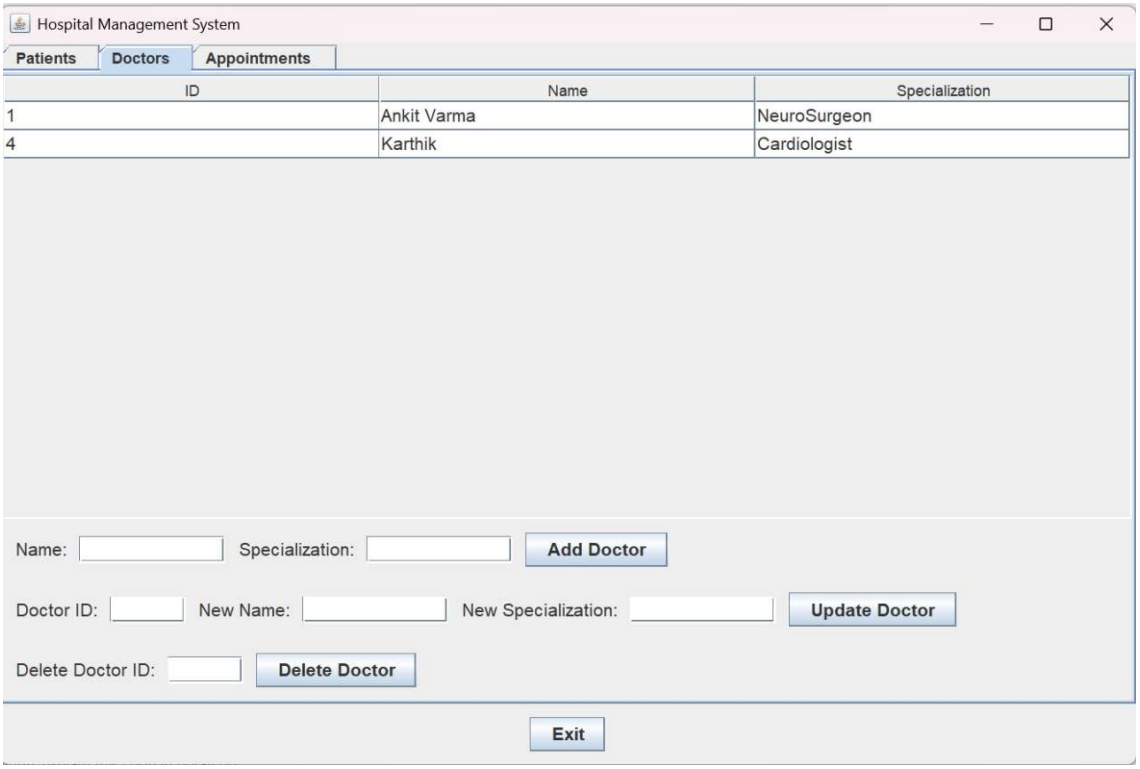
Update Patients



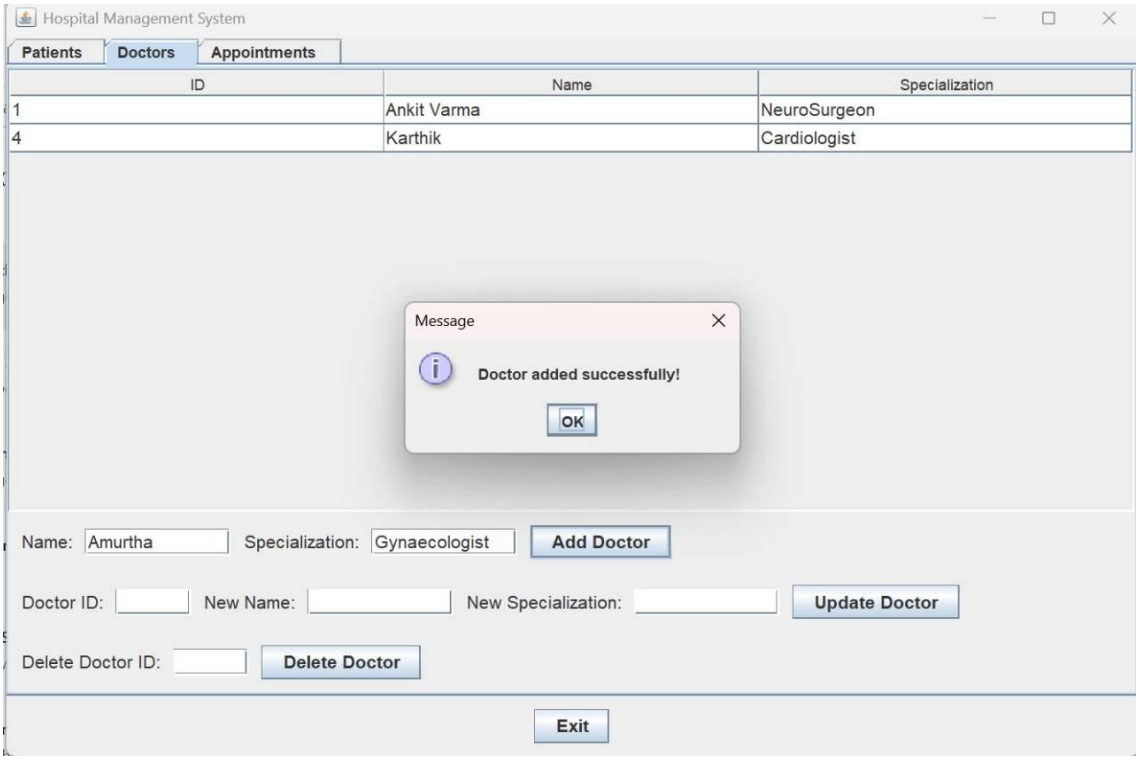
Delete Patients



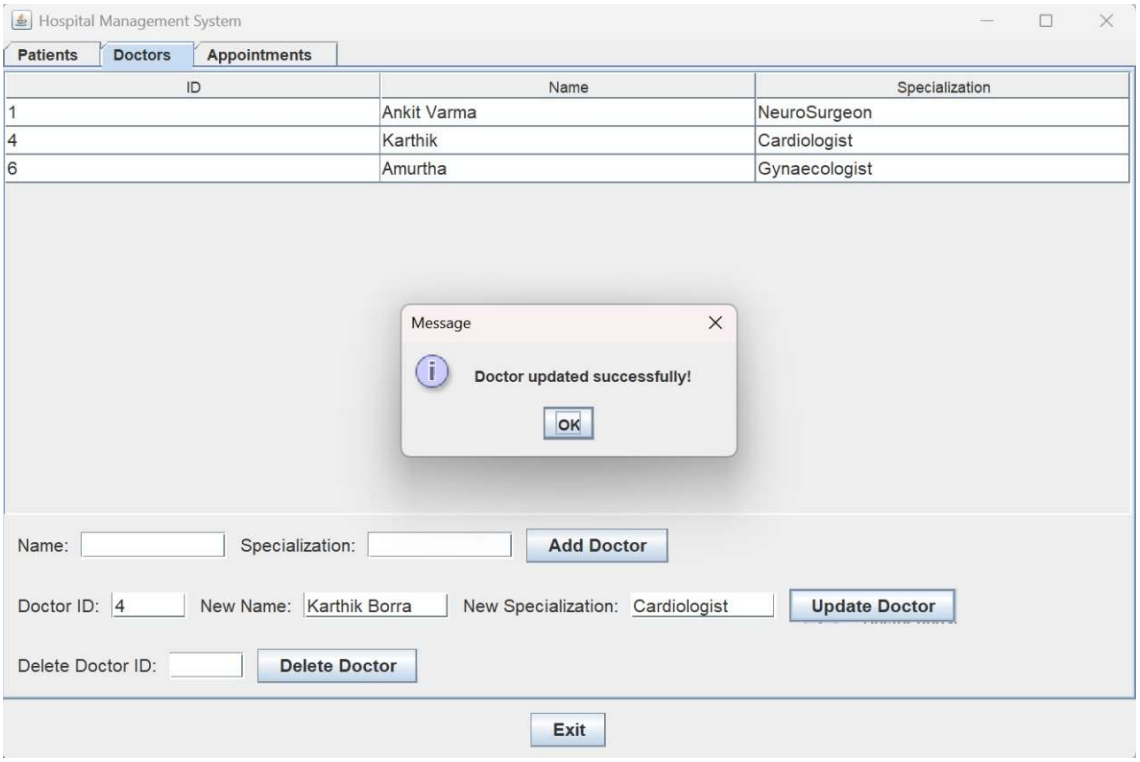
View Doctors



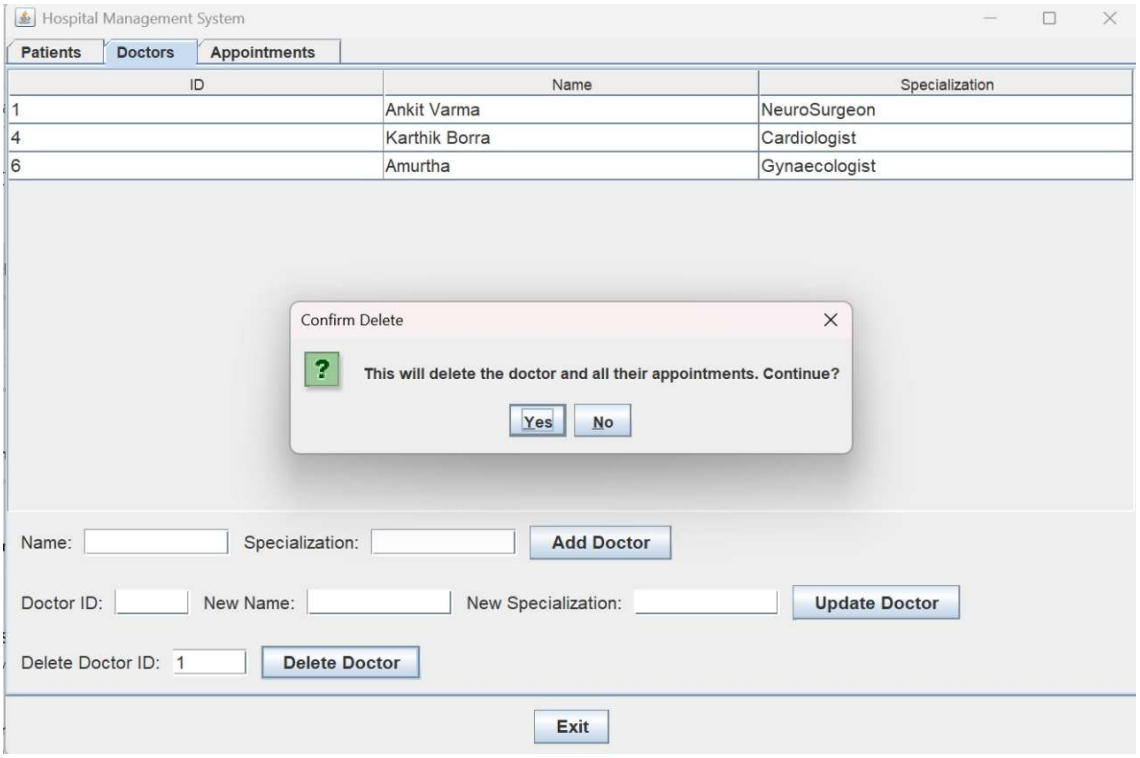
Add Doctors



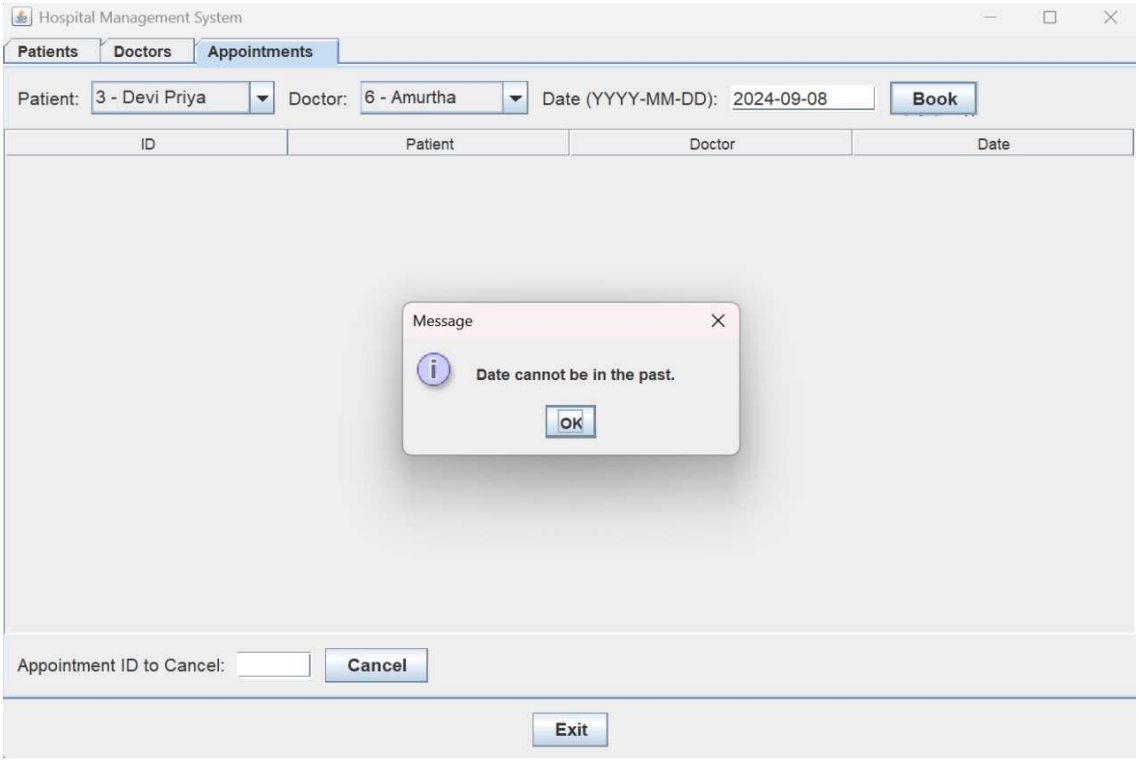
Update Doctors

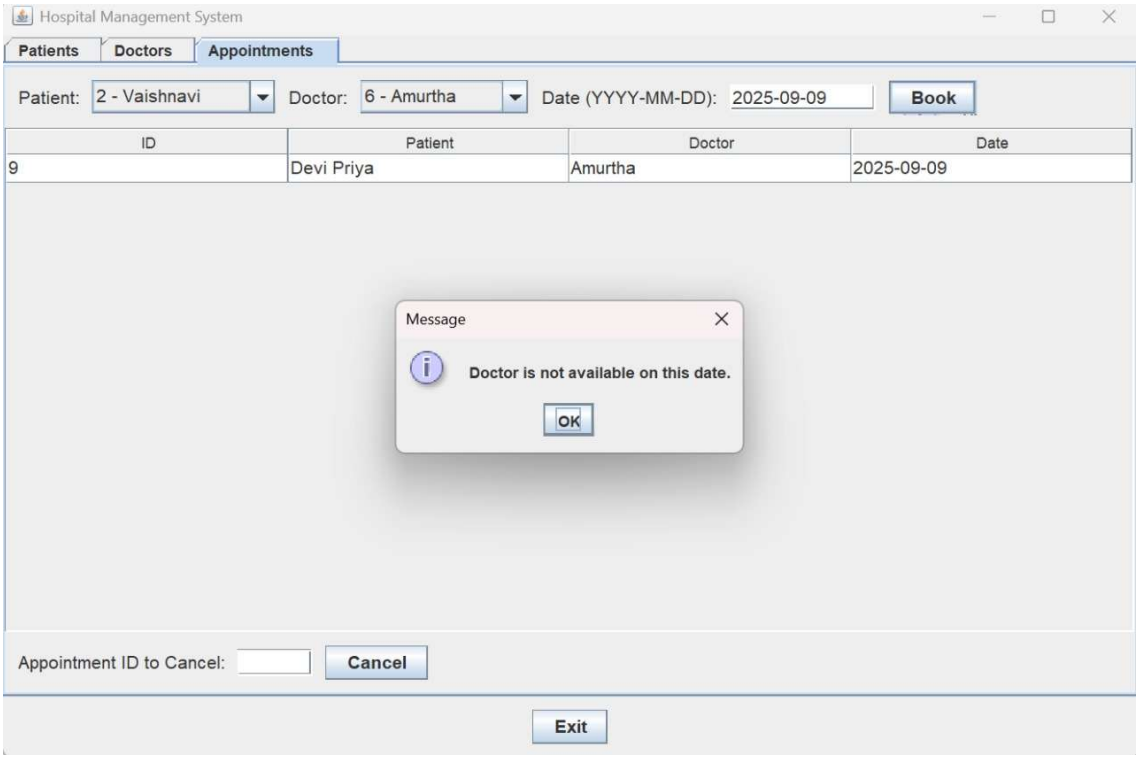


Delete Doctors

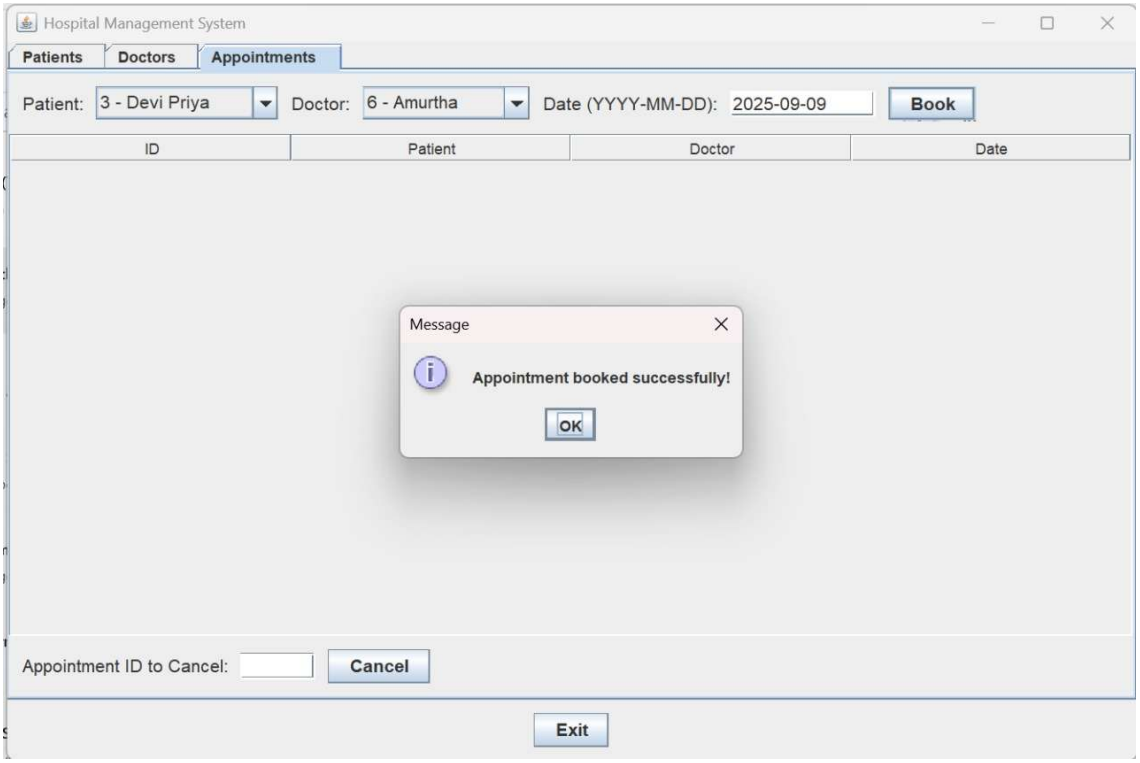


Appointments Constraints

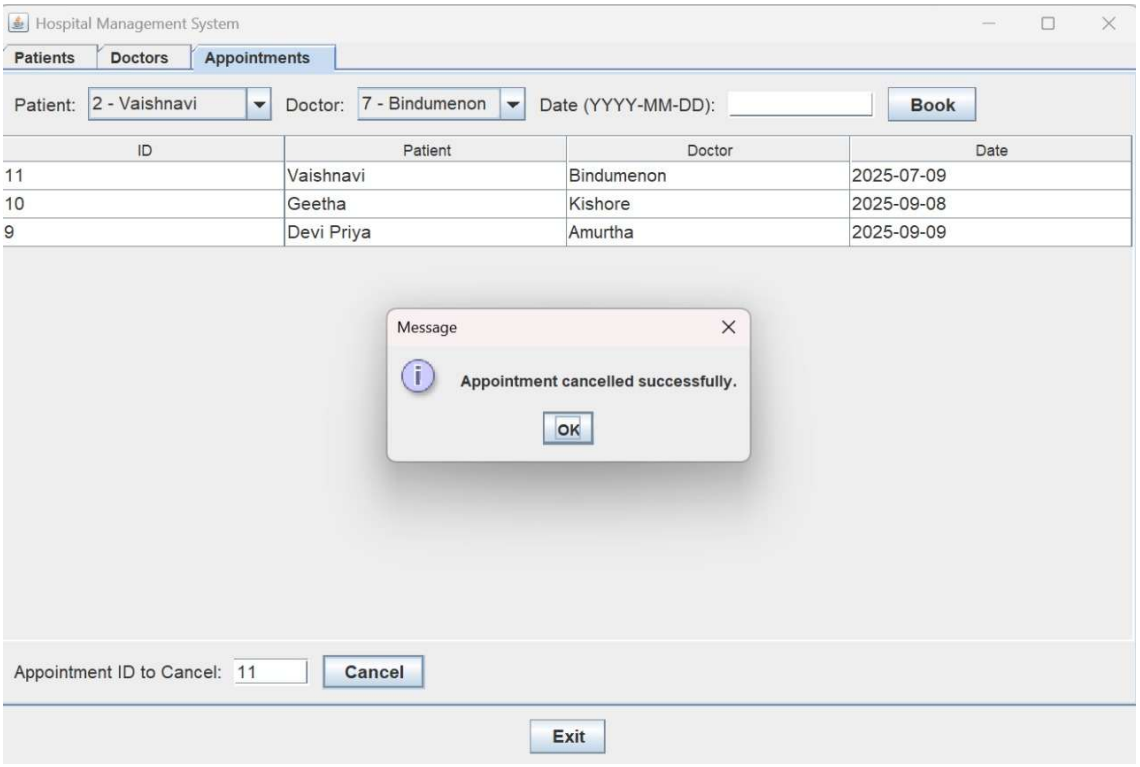




Book Appointment



Cancel Appointment



CONCLUSION

The **Hospital Management System (HMS)** successfully addresses the critical need for automated, efficient, and error-free healthcare administration. By leveraging Java Swing for an intuitive user interface and MySQL for reliable data management, the system provides a seamless workflow for handling patient records, doctor details, and appointment scheduling. Key strengths include real-time data synchronization, robust input validation, and secure database operations, ensuring data integrity and system reliability. This project demonstrates how well-designed software solutions can significantly improve operational efficiency in healthcare institutions while reducing manual workload and minimizing errors. The HMS serves as a foundational platform that bridges the gap between healthcare providers and digital management tools, ultimately contributing to better patient care and streamlined hospital operations.

FUTURE WORK

1. Role-Based Access Control (RBAC)

- Implement multi-tier user authentication (Admin, Doctor, Receptionist) with secure login credentials to restrict access based on roles.

2. Billing & Pharmacy Integration

- Extend functionality to include patient billing, insurance claims, and prescription management for a comprehensive hospital solution.

3. Advanced Reporting & Analytics

- Integrate data visualization tools (e.g., charts, dashboards) to generate performance reports, patient trends, and resource utilization statistics.

4. Mobile & Web Compatibility

- Develop cross-platform version (Android/iOS app or web portal) to allow doctors and patients to access records on-the-go.

5. Multi-Language & Accessibility Support

- Add localization features (multiple languages) and accessibility options (screen readers, high-contrast UI) for inclusive usability.

REFERENCES

https://docs.oracle.com/javase/tutorial/uiswing/.
https://docs.oracle.com/javase/8/docs/api/.
https://www.tutorialspoint.com/java_swing/index.htm.
https://www.technotalkative.com/java-swing- layouts/.