**Hospital Management System Project Report**

**SOFTWARE ENGINEERING REPORT**

|  |  |
| --- | --- |
| Name:Vedashree Bhandigare | B.Tech CE |
| DIV-B | A166 |

**Introduction**

The Hospital Management System (HMS) is a software application designed to manage the operations of a hospital, including patient and doctor management. The system provides functionalities to add, display, and delete patient and doctor records through a user-friendly graphical user interface (GUI). This project utilizes various software engineering principles and technologies to ensure a robust, efficient, and maintainable application.

## Technologies Used

### 1. **Programming Language: Java-Eclipse**

Java is used for its platform independence, object-oriented features, and robust library support for GUI and database operations. Eclipse was used as the Integrated Development Environment (IDE) for writing and managing the Java code

### 2. **GUI Framework: Java Swing**

Java Swing is used to create the graphical user interface, providing a rich set of widgets and layout management capabilities.

### 3. **Database: MySQL**

MySQL is chosen for its reliability, scalability, and wide acceptance in the industry. The database stores patient and doctor records securely.

### 4. **Date Picker: JDateChooser**

JDateChooser, a third-party library, is used for date selection in the AddPatientPage to ensure consistent and error-free date input.

### **Project Development Steps**

#### 1. Planning

* Define the scope and requirements of the Hospital Management System.
* Identify the key features: adding, displaying, and deleting patient and doctor records.
* Choose the technology stack: Java for application development, Swing for GUI, and MySQL for the database.

#### 2. Design

* Design the database schema for patients and doctors.
* Create wireframes and mockups for the GUI.
* Plan the class structure and interactions between different components.

#### 3. Coding and Implementation

* Set up the development environment using Eclipse IDE.
* Implement the database connection and CRUD operations for patients and doctors.
* Develop the GUI using Java Swing.
* Integrate the GUI with the database operations.

#### 4. Testing

* Perform unit testing on individual methods and classes to ensure they function correctly.
* Conduct integration testing to verify that different components work together seamlessly.
* Test the application with sample data to ensure it meets the requirements.

## Software Engineering Principles Applied

### 1. **Modularity**

The system is divided into several modules, each handling different functionalities:

* FrontPage: Main interface for user interactions.
* AddPatientPage: Handles adding patient records.
* AddDoctorPage: Handles adding doctor records.
* deletePatientPage: Handles deleting patient records.
* deleteDoctorPage: Handles deleting doctor records.

### 2. **Encapsulation**

The data fields and methods are encapsulated within their respective classes. Private fields and methods are used to protect the data and ensure proper encapsulation. Each class encapsulates its data and provides methods to interact with it, ensuring data integrity and security.

### 3. **Abstraction**

Complex operations such as database connectivity and data manipulation are abstracted into methods, simplifying the code and improving readability.

### 4. **Code Reusability**

The code structure encourages reuse of components, such as the database connection setup, labels, text fields, and action listeners for buttons. The code for displaying records is also designed to be reusable.

### 5. **User-Friendly Interface**

The system uses Java Swing to create a simple and intuitive GUI, ensuring ease of use for non-technical users.

### 6. **Error Handling**

Exception handling is implemented to manage database connection errors and SQL exceptions, enhancing the system's robustness.

7. **Maintainability**

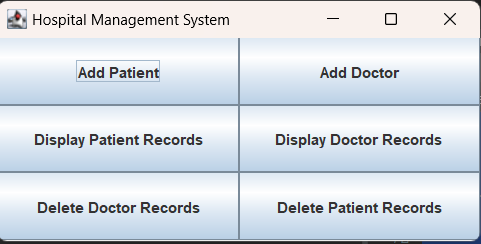
The code is written in a clear and organized manner, following standard Java conventions. This makes it easier to maintain and extend in the future.

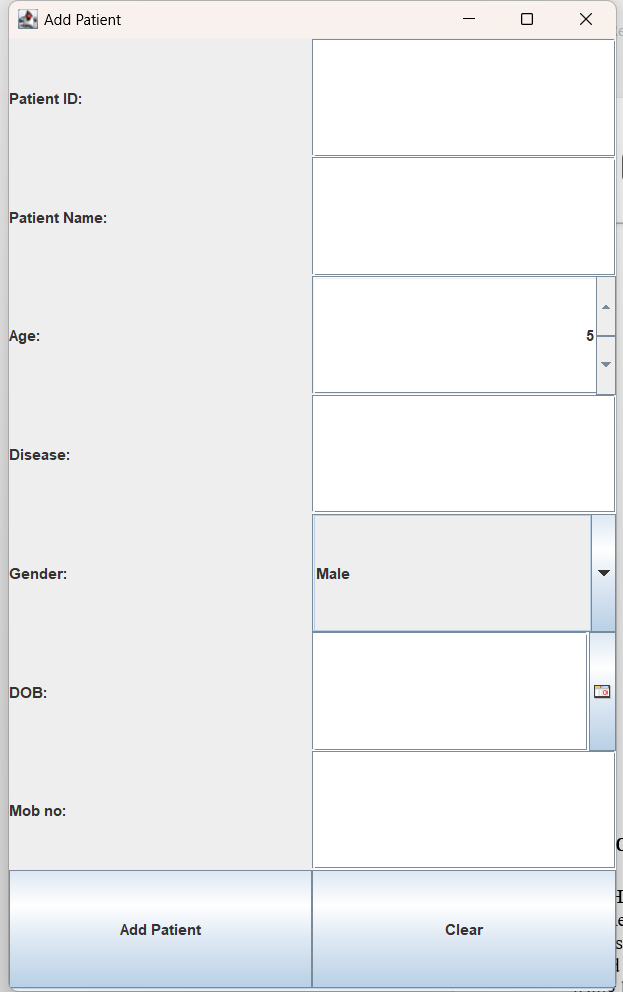
### **Drawbacks and Pitfalls of Using Eclipse:**

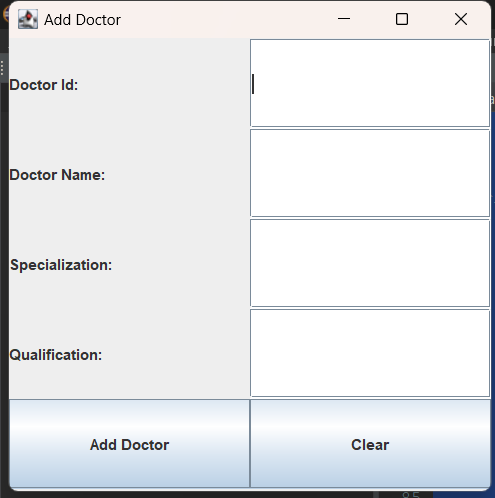
While Eclipse is a powerful IDE, there were some drawbacks encountered during the development of this project:

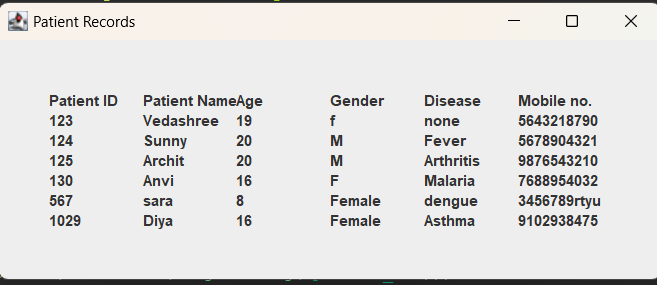
1. **Performance Issues**: Eclipse can be slow, especially when dealing with large projects. The IDE sometimes becomes unresponsive, which can hinder productivity.
2. **Complex Configuration**: Setting up the environment and configuring external libraries (like JDateChooser) can be cumbersome and time-consuming.
3. **Memory Consumption**: Eclipse consumes a lot of memory, which can be an issue on systems with limited resources.

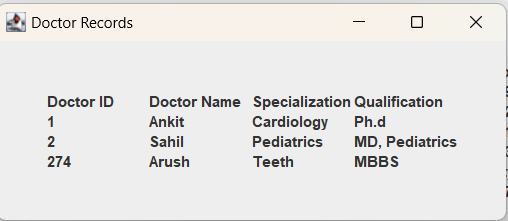
**Output**:

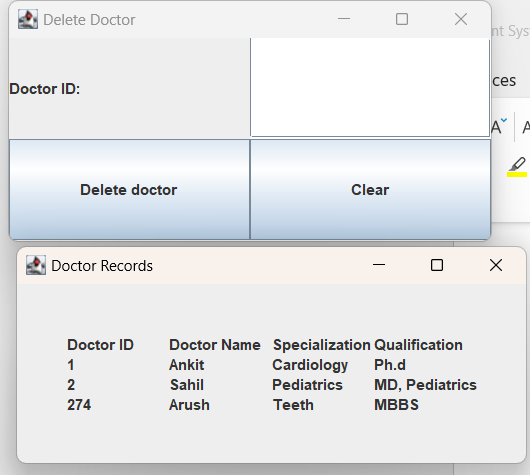


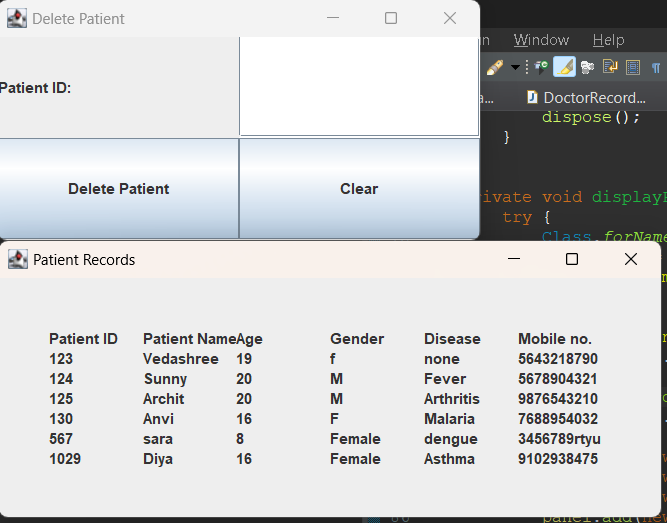












### **Conclusion:**

The Hospital Management System project demonstrates the application of software engineering principles to create a modular, maintainable, and functional application. While Eclipse was a useful tool for development, its performance and configuration complexities posed challenges. Future improvements could involve switching to a more efficient IDE and using build tools to streamline the development process. The systematic approach of planning, design, coding, and testing ensured a robust and reliable system.