Name : NISHIDHA MANOHAR JADHAV

Student ID : AF04948282

Batch Code : ANP-D1253

Project Name : Hospital Management System

Project Guide : Ms. Maseera Jamal Shaikh

Objectives :

1. To create a Java Swing-based GUI application for managing hospital-related data.
2. To integrate the application with a MySQL database for secure and persistent storage of patient, doctor, appointment, and billing records.
3. To implement CRUD (Create, Read, Update, Delete) operations for key hospital entities such as patients, doctors, staff, appointments, and treatments.
4. To demonstrate the use of MVC (Model-View-Controller) architecture for better separation of concerns and maintainability.
5. To provide a user-friendly and intuitive interface for hospital staff to efficiently manage day-to-day operations.
6. To ensure data validation, input checks, and appropriate error handling to maintain data integrity and reliability.
7. To enable search and filtering functionalities to quickly access relevant records.
8. To maintain logs or reports for activities such as patient visits, billing history, and doctor availability.

Problem Statements :

There is a need for a computerized Hospital Management System (HMS) that can automate and streamline these operations. The system should offer a centralized platform to manage patient information, doctor details, appointments, medical history, billing, and other administrative tasks. A well-designed HMS will improve efficiency, reduce human error, and enhance the quality of healthcare services by providing accurate and timely information to hospital staff.

Data Description :

The application manages the following data:

1. Patient Table

- Stores registered patient details.
- Fields: Patient_ID, Full_Name, Age, Gender, Contact_Number, Address, Email, Medical_History

2. Doctor Table

- Contains information about hospital doctors.
- Fields: Doctor_ID, Full_Name, Specialization, Contact_Number, Email, Availability_Status

3. Appointment Table

- Manages appointments between patients and doctors.
- Fields: Appointment_ID, Patient_ID, Doctor_ID, Appointment_Date, Appointment_Time, Status

4. Staff Table

- Stores details of non-doctor hospital staff (nurses, admin, etc.).
- Fields: Staff_ID, Full_Name, Role, Contact_Number, Email, Shift_Timing

5. Room Table

- Tracks room types, availability, and patient assignment.
- Fields: Room_ID, Room_Type, Availability_Status, Assigned_Patient_ID

6. Billing Table

- Maintains billing and payment info for patients.

- Fields: Bill_ID, Patient_ID, Total_Amount, Date_Issued, Payment_Status

7. Medical Records Table

- Stores patient diagnosis, treatments, and prescriptions.
- Fields: Record_ID, Patient_ID, Doctor_ID, Diagnosis, Treatment, Prescriptions, Date

Data Models :

Columns:

- **Patient_ID**
  - Data Type: INT
  - Constraints: PRIMARY KEY, AUTO_INCREMENT
  - Description: Unique identifier for each patient.
- **Full_Name**
  - Data Type: VARCHAR(100)
  - Constraints: NOT NULL
  - Description: Full name of the patient.
- **Age**
  - Data Type: INT
  - Constraints: NOT NULL
  - Description: Age of the patient.
- **Gender**
  - Data Type: VARCHAR(10)
  - Constraints: NOT NULL
  - Description: Gender of the patient.
- **Contact_Number**
  - Data Type: VARCHAR(15)
  - Constraints: NOT NULL
  - Description: Phone number of the patient.

- **Address**
  - Data Type: TEXT
  - Constraints: NULL
  - Description: Residential address.
- **Email**
  - Data Type: VARCHAR(100)
  - Constraints: NULL
  - Description: Email address of the patient.
- **Medical_History**
  - Data Type: TEXT
  - Constraints: NULL
  - Description: Past or ongoing medical conditions.

Purpose:

- Stores complete personal and medical information of patients.
- Ensures uniqueness with Patient_ID and data accuracy through constraints.

Approach:

1. Model (Data & Logic Layer)

- Contains Java classes representing core entities:
  - Patient.java, Doctor.java, Appointment.java, Room.java, Billing.java, MedicalRecord.java, Staff.java
- Each class holds attributes and methods related to that entity.

2. View (User Interface)

- Developed using Java Swing.
- Provides forms and dialogs for:
  - Registering patients and staff
  - Scheduling appointments
  - Viewing billing and room status

- Examples: PatientView.java, AppointmentView.java, DashboardView.java

3. Controller (Interaction Layer)

- Handles communication between the Model and View.
- Captures user actions from the UI and updates data accordingly.

4. Database Layer

- Uses **JDBC** for MySQL connectivity.
- DAO (Data Access Object) classes handle all CRUD operations:

Project Result :

I. Patients, doctors, appointments, billing, staff, and room assignments can be **efficiently managed** through a user-friendly interface.
II. All **CRUD operations** (Create, Read, Update, Delete) are fully functional for each module.
III. The system ensures **data consistency and validation**, minimizing manual errors.
IV. Real-time interaction with the MySQL database was implemented using **JDBC**.
V. The modular structure allows for easy future enhancements like report generation, user login roles, or cloud storage.

Conclusion :

The Hospital Management System effectively streamlines hospital operations by managing patients, doctors, appointments, billing, and medical records through a user-friendly interface. Developed using Java Swing and MySQL with an MVC architecture, the system ensures organized code, smooth data handling, and easy maintenance. It reduces manual errors, improves data accuracy, and provides a strong base for future enhancements like login systems or analytics features.