KARPAGAM
COLLEGE OF ENGINEERING
Rediscover | Refine | Redefine
(Autonomous)

21PE04/21YE04 /21DE04/21SE04/21FE04

ADVANCED JAVA PROGRAMMING

HOSPITAL MANAGEMENT SYSTEM

SEMSTER IV

2024

JAYARAJ.J

717822P221

CSE

Abstract:

- The Hospital Management System facilitates efficient management of patient records and diagnoses. Utilizing Java and MySQL, it enables seamless data entry and retrieval. Patients' details, including ID, name, birth date, gender, address, and contact, are stored securely. The system also offers diagnosis recommendations based on entered diseases such as COVID-19, Ulcer, and Diabetes. With its user-friendly interface, the system ensures streamlined operations and improved patient care.

Table of Contents:

1. Introduction:

- Efficient Patient Record Management: The system facilitates the creation and maintenance of patient records, including essential details such as patient ID, name, date of birth, gender, address, and phone number.
- Enhanced Data Accessibility: By centralizing patient information in a database, healthcare professionals can quickly access patient records, enabling informed decision-making and providing timely medical assistance.
- Streamlined Patient Registration: With an intuitive interface, the system simplifies the patient registration process, allowing for the seamless entry of patient details.
- Disease Identification and Recommendations: Upon registering a patient, the system identifies the disease based on user input and provides relevant recommendations for further medical care or treatment.

2. System Analysis:

- **Requirements Analysis:**
- **Patient Record Management:**
- The system should allow for the creation, retrieval, updating, and deletion of patient records.
- Each patient record should include attributes such as patient ID, name, date of birth, gender, address, phone number, and disease.
- **User Interface:**
- The system should have a user-friendly interface for easy navigation and data entry.
- Input forms should prompt users to enter necessary patient details accurately.
- **Database Management:**
- The system should use a MySQL database to store patient records securely.
- Proper database schema design should be implemented to ensure data integrity and efficient retrieval.
- **Data Validation:**
- Input data should be validated to prevent errors and maintain data consistency.
- Validation rules should be applied to fields such as date of birth, gender, and phone number to ensure correctness.
- **Disease Identification:**
- The system should identify the disease based on user input and provide relevant recommendations or instructions.
- Disease identification should be accurate and support a variety of common medical conditions.
- Security and Access Control:

- Access to patient records should be restricted to authorized personnel only.
- User authentication mechanisms should be implemented to ensure data confidentiality and prevent unauthorized access.
- Specifications of the System:
- **Programming Language:** Java
- **Database Management System**: MySQL8
- **User Interface:** Command-line interface (CLI) for simplicity and ease of use.
- **Data Validation:** Input validation for all user-entered data to ensure data integrity.
- **Database Schema:**
- **Table:** Patients
- **Columns:** patient_id (INT), patient_name (VARCHAR), date_of_birth (DATE), gender (ENUM), address (VARCHAR), phone_number (VARCHAR), Disease (VARCHAR)
- Disease Identification:
- **Supported Diseases**: COVID-19, Ulcer, Diabetes (additional diseases can be added as needed)
- Recommendations: Specific instructions provided based on the identified disease.
- **Error Handling:** Comprehensive error handling to manage exceptions gracefully and provide meaningful error messages.
- Security: Basic security measures implemented within the system to ensure data privacy and integrity.

3. System Design:

- **Database Schema Design:**
- The database schema for hospital data management includes tables to store patient information and disease details. Here's a basic schema design:
- **Table: Patients**
- **Columns:**
- patient_id (INT, PRIMARY KEY): **Unique identifier for each patient**.
- patient_name (VARCHAR**): Name of the patient.**
- date_of_birth (DATE): **Date of birth of the patient.**
- gender (ENUM('Male', 'Female', 'Other')): **Gender of the patient.**
- address (VARCHAR): **Address of the patient.**
- phone_number (VARCHAR): **Phone number of the patient**.
- Disease (VARCHAR): **Disease diagnosed for the patient**.
- **Table: Diseases**
- **Columns:**
- disease_id (INT, PRIMARY KEY): **Unique identifier for each disease**.
- disease_name (VARCHAR): **Name of the disease.**
- description (TEXT): **Description of the disease and its symptoms.**
- This schema design allows for the storage of patient records along with their associated diseases, enabling efficient management and retrieval of hospital data.
- **Application Architecture Design:**
- The Hospital Management System follows a layered architecture, consisting of the following components.
- **Presentation Layer:**
- **User Interface:** Command-line interface (CLI) for user interaction.
- **Business Logic Layer:**
- Java classes responsible for handling business logic, including:

- Input validation
- Database operations (CRUD operations)
- Disease identification and recommendations
- **Data Access Layer:**
- Database connectivity and interaction through JDBC (Java Database Connectivity).
- Prepared statements for executing SQL queries and interacting with the MySQL database.
- **UML Diagrams:**
- Since the system architecture is relatively simple and the components are described above, UML diagrams such as Class Diagrams or Sequence Diagrams may not be necessary for this design. However, if needed, we can create basic diagrams to illustrate the relationships between classes or the flow of interactions between system components.

4. Implementation:

Environment Setup:

- **JDBC Setup**:
  - Ensure you have the JDBC driver for MySQL. You can download it from the MySQL 8.
  - Add the JDBC driver JAR file to your project's classpath.
- **Database Choice**:
  - MySQL8 is used for this project. Make sure you have MySQL8 installed on your system or have access to a MySQL server.

References:

- Sierra, K., & Bates, B. (2005). Head First Java. O'Reilly Media.
- MySQL Documentation. Retrieved from https://dev.mysql.com/doc/

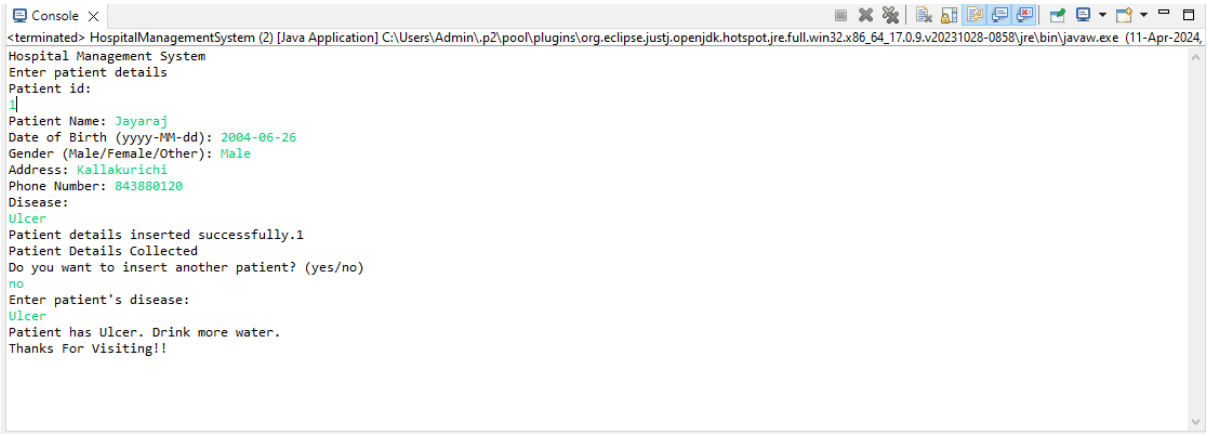
SourceCode:

```
                    package hospital;
        import java.sql.*;
        import java.util.Scanner;
public class HospitalManagementSystem {
   static String url="jdbc:mysql://localhost:1009/hospital";
   static String username="root";
   static String password="8826";
   static Scanner scanner = new Scanner(System.in);
   public static void main(String[] args) {
      try {
         Class.forName("com.mysql.cj.jdbc.Driver");
      } catch (ClassNotFoundException e) {
         e.printStackTrace();}
      try (Connection connection = DriverManager.getConnection(url,username,password)) {
         System.out.println("Hospital Management System");
         Statement statement=connection.createStatement();
         String query1="CREATE TABLE IF NOT EXISTS Patients (\r\n"
               + "    patient_id INT PRIMARY KEY,\r\n"
               + "    patient_name VARCHAR(100) NOT NULL,\r\n"
               + "    date_of_birth DATE NOT NULL,\r\n"
               + "    gender ENUM('Male', 'Female', 'Other') NOT NULL,\r\n"
               + "    address VARCHAR(255) NOT NULL,\r\n"
               + "    phone_number VARCHAR(10) NOT NULL\r\n"
               + ");\r\n"
               + "";
         statement.executeUpdate(query1);
         // Loop to insert patient details
         while (true) {
            System.out.println("Enter patient details");
            System.out.println("Patient id:");
            int patient_id = scanner.nextInt();
            scanner.nextLine(); // Consume newline character
            System.out.print("Patient Name: ");
            String patientName = scanner.nextLine();
            System.out.print("Date of Birth (yyyy-MM-dd): ");
            String dateOfBirth = scanner.next();
            System.out.print("Gender (Male/Female/Other): ");
            String gender = scanner.next();
            System.out.print("Address: ");
            String address = scanner.next();
            System.out.print("Phone Number: ");
            String phoneNumber = scanner.next();
            System.out.println("Disease:");
            String Disease=scanner.next();
            // SQL query to insert user input into Patients table
            String insertPatientQuery = "INSERT INTO Patients (patient_id, patient_name, date_of_birth, gender, address,
phone_number,Disease) VALUES (?, ?, ?, ?, ?, ?,?)";
            PreparedStatement statement1 = connection.prepareStatement(insertPatientQuery);
            statement1.setInt(1, patient_id);
```

```
            statement1.setString(2, patientName);
            statement1.setString(3, dateOfBirth);
            statement1.setString(4, gender);
            statement1.setString(5, address);
            statement1.setString(6, phoneNumber);
            statement1.setString(7, Disease);
            // Execute the insert statement
            int rowsInserted = statement1.executeUpdate();
            if (rowsInserted > 0) {
               System.out.println("Patient details inserted successfully."+rowsInserted);
               System.out.println("Patient Details Collected");
            } else {
               System.out.println("Failed to insert patient details.");
            }
            // Ask the user if they want to insert another patient
            System.out.println("Do you want to insert another patient? (yes/no)");
            String choice = scanner.next();
            if (!choice.equalsIgnoreCase("yes")) {
               break; // Exit the loop if the user does not want to insert another patient
            }
         }
         System.out.println("Enter patient's disease:");
         String disease = scanner.next();
         switch (disease) {
            case "COVID-19":
               System.out.println("Patient has COVID-19. Follow quarantine protocols.");
               System.out.println("Thanks For Visiting!!");
               break;
            case "Ulcer":
               System.out.println("Patient has Ulcer. Drink more water.");
               System.out.println("Thanks For Visiting!!");
               break;
            case "Diabetes":
               System.out.println("Patient has Diabetes. Monitor blood sugar levels regularly.");
               System.out.println("Thanks For Visiting!!");
               break;
            // Add more cases as needed
            default:
               System.out.println("Unknown disease. Consult a healthcare professional.");
               System.out.println("Thanks For Visiting!!");
         }
      } catch (SQLException e) {
         e.printStackTrace();
       } finally {
         scanner.close();       }
}
}
```

OUTPUT:

IN JAVA ECLIPSE CONSOLE:



IN SQL WORKBENCH :