Idea

Mysql connector

```
<component name="libraryTable">
  <library name="mysql-connector-j-8.1.0">
    <CLASSES>
      <root
url="jar://$USER_HOME$/Downloads/mysql-connector-j-8.1.0/mysql-connector-j-8.1.0/mysql-co
nnector-j-8.1.0.jar!/" />
    </CLASSES>
    <JAVADOC />
    <SOURCES />
  </library>
</component>
```

Git ignore

```
# Default ignored files
/shelf/
/workspace.xml
```

Misc.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project version="4">
  <component name="ProjectRootManager" version="2" languageLevel="JDK_20"
default="true" project-jdk-name="openjdk-20" project-jdk-type="JavaSDK">
    <output url="file://$PROJECT_DIR$/out" />
  </component>
</project>
```

Modules.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project version="4">
  <component name="ProjectModuleManager">
    <modules>
      <module fileurl="file://$PROJECT_DIR$/Hospital Management System.iml"
filepath="$PROJECT_DIR$/Hospital Management System.iml" />
    </modules>
  </component>
</project>
```

Doctor.java

```
 package HospitalManagementSystem;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Scanner;

public class Doctor {
    private Connection connection;

    public Doctor(Connection connection){
        this.connection = connection;
    }

    public void viewDoctors(){
        String query = "select * from doctors";
        try{
            PreparedStatement preparedStatement = connection.prepareStatement(query);
            ResultSet resultSet = preparedStatement.executeQuery();
            System.out.println("Doctors: ");
            System.out.println("+------------+--------------------+------------------+");
            System.out.println("| Doctor Id  | Name               | Specialization   |");
            System.out.println("+------------+--------------------+------------------+");
            while(resultSet.next()){
                int id = resultSet.getInt("id");
                String name = resultSet.getString("name");
                String specialization = resultSet.getString("specialization");
                System.out.printf("| %-10s | %-18s | %-16s |\n", id, name, specialization);
                System.out.println("+------------+--------------------+------------------+");
            }

        }catch (SQLException e){
            e.printStackTrace();
        }
    }

    public boolean getDoctorById(int id){
        String query = "SELECT * FROM doctors WHERE id = ?";
        try{
            PreparedStatement preparedStatement = connection.prepareStatement(query);
            preparedStatement.setInt(1, id);
            ResultSet resultSet = preparedStatement.executeQuery();
            if(resultSet.next()){
                return true;
            }else{
                return false;
            }
        }catch (SQLException e){
            e.printStackTrace();
        }
        return false;
    }
}
```

<u>HospitalManagementSystem.java</u>

```
package HospitalManagementSystem;

import java.sql.*;
import java.util.Scanner;

public class HospitalManagementSystem {
    private static final String url = "jdbc:mysql://localhost:3306/hospital";
    private static final String username = "root";
    private static final String password = "Admin@123";

    public static void main(String[] args) {
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
        }catch (ClassNotFoundException e){
            e.printStackTrace();
        }
        Scanner scanner = new Scanner(System.in);
        try{
            Connection connection = DriverManager.getConnection(url, username, password);
            Patient patient = new Patient(connection, scanner);
            Doctor doctor = new Doctor(connection);
            while(true){
                System.out.println("HOSPITAL MANAGEMENT SYSTEM ");
                System.out.println("1. Add Patient");
                System.out.println("2. View Patients");
                System.out.println("3. View Doctors");
                System.out.println("4. Book Appointment");
                System.out.println("5. Exit");
                System.out.println("Enter your choice: ");
                int choice = scanner.nextInt();

                switch(choice){
                    case 1:
                        // Add Patient
                        patient.addPatient();
                        System.out.println();
                        break;
                    case 2:
                        // View Patient
                        patient.viewPatients();
                        System.out.println();
                        break;
                    case 3:
                        // View Doctors
                        doctor.viewDoctors();
                        System.out.println();
                        break;
                    case 4:
                        // Book Appointment
                        bookAppointment(patient, doctor, connection, scanner);
                        System.out.println();
                        break;
                    case 5:
                        System.out.println("THANK YOU! FOR USING HOSPITAL MANAGEMENT
SYSTEM!!");
                        return;
                    default:
                        System.out.println("Enter valid choice!!!");
                        break;
                }

            }

        }catch (SQLException e){
            e.printStackTrace();
        }
    }


    public static void bookAppointment(Patient patient, Doctor doctor, Connection connection,
Scanner scanner){
        System.out.print("Enter Patient Id: ");
        int patientId = scanner.nextInt();
        System.out.print("Enter Doctor Id: ");
        int doctorId = scanner.nextInt();
        System.out.print("Enter appointment date (YYYY-MM-DD): ");
        String appointmentDate = scanner.next();
        if(patient.getPatientById(patientId) && doctor.getDoctorById(doctorId)){
            if(checkDoctorAvailability(doctorId, appointmentDate, connection)){
                String appointmentQuery = "INSERT INTO appointments(patient_id, doctor_id,
appointment_date) VALUES(?, ?, ?)";
```

```
                try {
                    PreparedStatement preparedStatement =
connection.prepareStatement(appointmentQuery);
                    preparedStatement.setInt(1, patientId);
                    preparedStatement.setInt(2, doctorId);
                    preparedStatement.setString(3, appointmentDate);
                    int rowsAffected = preparedStatement.executeUpdate();
                    if(rowsAffected>0){
                        System.out.println("Appointment Booked!");
                    }else{
                        System.out.println("Failed to Book Appointment!");
                    }
                }catch (SQLException e){
                    e.printStackTrace();
                }
            }else{
                System.out.println("Doctor not available on this date!!");
            }
        }else{
            System.out.println("Either doctor or patient doesn't exist!!!");
        }
    }

    public static boolean checkDoctorAvailability(int doctorId, String appointmentDate,
Connection connection){
        String query = "SELECT COUNT(*) FROM appointments WHERE doctor_id = ? AND
appointment_date = ?";
        try{
            PreparedStatement preparedStatement = connection.prepareStatement(query);
            preparedStatement.setInt(1, doctorId);
            preparedStatement.setString(2, appointmentDate);
            ResultSet resultSet = preparedStatement.executeQuery();
            if(resultSet.next()){
                int count = resultSet.getInt(1);
                if(count==0){
                    return true;
                }else{
                    return false;
                }
            }
        } catch (SQLException e){
            e.printStackTrace();
        }
        return false;
    }
}
```

Patient.java

```
package HospitalManagementSystem;

import java.sql.*;
import java.util.Scanner;

public class Patient {
    private Connection connection;
    private Scanner scanner;

    public Patient(Connection connection, Scanner scanner){
        this.connection = connection;
        this.scanner = scanner;
    }

    public void addPatient(){
        System.out.print("Enter Patient Name: ");
        String name = scanner.next();
        System.out.print("Enter Patient Age: ");
        int age = scanner.nextInt();
        System.out.print("Enter Patient Gender: ");
        String gender = scanner.next();

        try{
            String query = "INSERT INTO patients(name, age, gender) VALUES(?, ?, ?)";
            PreparedStatement preparedStatement = connection.prepareStatement(query);
            preparedStatement.setString(1, name);
            preparedStatement.setInt(2, age);
            preparedStatement.setString(3, gender);
            int affectedRows = preparedStatement.executeUpdate();
            if(affectedRows>0){
                System.out.println("Patient Added Successfully!!");
            }else{
                System.out.println("Failed to add Patient!!");
            }

        }catch (SQLException e){
            e.printStackTrace();
        }
    }
    public void viewPatients(){
        String query = "select * from patients";
        try{
            PreparedStatement preparedStatement = connection.prepareStatement(query);
            ResultSet resultSet = preparedStatement.executeQuery();
            System.out.println("Patients: ");
            System.out.println("+------------+--------------------+----------+------------+");
            System.out.println("| Patient Id | Name               | Age      | Gender     |");
            System.out.println("+------------+--------------------+----------+------------+");
            while(resultSet.next()){
                int id = resultSet.getInt("id");
                String name = resultSet.getString("name");
                int age = resultSet.getInt("age");
                String gender = resultSet.getString("gender");
                System.out.printf("| %-10s | %-18s | %-8s | %-10s |\n", id, name, age, gender);
                System.out.println("+------------+--------------------+----------+------------+");
            }

        }catch (SQLException e){
            e.printStackTrace();
        }
    }

    public boolean getPatientById(int id){
        String query = "SELECT * FROM patients WHERE id = ?";
        try{
            PreparedStatement preparedStatement = connection.prepareStatement(query);
            preparedStatement.setInt(1, id);
            ResultSet resultSet = preparedStatement.executeQuery();
            if(resultSet.next()){
                return true;
            }else{
                return false;
            }
        }catch (SQLException e){
            e.printStackTrace();
        }
        return false;
    }

}
```

3)git ignore

```
### IntelliJ IDEA ###
out/
```

```
!**/src/main/**/out/
!**/src/test/**/out/

### Eclipse ###
.apt_generated
.classpath
.factorypath
.project
.settings
.springBeans
.sts4-cache
bin/
!**/src/main/**/bin/
!**/src/test/**/bin/

### NetBeans ###
/nbproject/private/
/nbbuild/
/dist/
/nbdist/
/.nb-gradle/

### VS Code ###
.vscode/

### Mac OS ###
.DS_Store
```

4)hospital management system

```
 <?xml version="1.0" encoding="UTF-8"?>
<module type="JAVA_MODULE" version="4">
  <component name="NewModuleRootManager" inherit-compiler-output="true">
    <exclude-output />
    <content url="file://$MODULE_DIR$">
      <sourceFolder url="file://$MODULE_DIR$/src" isTestSource="false" />
    </content>
    <orderEntry type="inheritedJdk" />
    <orderEntry type="sourceFolder" forTests="false" />
    <orderEntry type="library" name="mysql-connector-j-8.1.0" level="project" />
  </component>
</module>
```

Perfect! Here are the important Java + MySQL syntaxes you need to understand and write projects like your Hospital Management System, grouped by topics 👇

✅ 1. Basic JDBC Connection Syntax

```
import java.sql.*;

String url = "jdbc:mysql://localhost:3306/database_name";
String username = "root";
String password = "your_password";

Connection conn = DriverManager.getConnection(url, username, password);
```

🔗 W3Schools JDBC

✅ 2. Load MySQL JDBC Driver (Optional for newer versions)

```
Class.forName("com.mysql.cj.jdbc.Driver");
```

🔗 GFG – JDBC Driver

✅ 3. PreparedStatement Syntax (Safe SQL Execution)

```
String query = "INSERT INTO patients(name, age, gender) VALUES (?, ?, ?)";
PreparedStatement ps = conn.prepareStatement(query);
ps.setString(1, "Seenu");
ps.setInt(2, 20);
ps.setString(3, "Female");
ps.executeUpdate();
```

🔗 GFG – PreparedStatement

✅ 4. Select Query with ResultSet

```
String query = "SELECT * FROM doctors";
PreparedStatement ps = conn.prepareStatement(query);
ResultSet rs = ps.executeQuery();

while(rs.next()) {
    int id = rs.getInt("id");
    String name = rs.getString("name");
    String specialization = rs.getString("specialization");
    System.out.println(id + " - " + name + " - " + specialization);
}
```

✅ 5. Scanner Input in Java

```
import java.util.Scanner;

Scanner sc = new Scanner(System.in);
System.out.print("Enter name: ");
String name = sc.nextLine();

System.out.print("Enter age: ");
int age = sc.nextInt();
```

🔗 Javatpoint – Scanner

✅ 6. Creating a Class with Constructor

```
public class Patient {
    private Connection conn;

    public Patient(Connection conn) {
        this.conn = conn;
    }
}
```

✅ 7. Menu-Driven Console Program

```
while(true) {
    System.out.println("1. Add Patient\n2. Exit");
    int choice = sc.nextInt();

    switch(choice) {
        case 1: addPatient(); break;
        case 2: System.exit(0); break;
        default: System.out.println("Invalid choice");
    }
}
```

🔗 GFG – Java Menu Program

✅ 8. Check Record Exists Using SELECT

```
String query = "SELECT * FROM patients WHERE id = ?";
PreparedStatement ps = conn.prepareStatement(query);
ps.setInt(1, 3);
ResultSet rs = ps.executeQuery();

if (rs.next()) {
    System.out.println("Patient Found");
} else {
    System.out.println("Patient Not Found");
}
```

✅ 9. Method Reuse Syntax

```
public boolean getPatientById(int id) {
    // SQL logic here
    return true or false;
}
```

Call it from main:

```
if (patient.getPatientById(2)) {
    System.out.println("Exists");
```

}

✅ 10. Create Table SQL (for setup in MySQL Workbench)

```
CREATE TABLE patients (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50),
    age INT,
    gender VARCHAR(10)
);

CREATE TABLE doctors (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50),
    specialization VARCHAR(50)
);

CREATE TABLE appointments (
    id INT AUTO_INCREMENT PRIMARY KEY,
    patient_id INT,
    doctor_id INT,
    appointment_date DATE,
    FOREIGN KEY (patient_id) REFERENCES patients(id),
    FOREIGN KEY (doctor_id) REFERENCES doctors(id)
);
```

🔗 W3Schools – SQL Create Table

✨ BONUS: Display Table Formatting in Console

```
System.out.printf("| %-10s | %-15s | %-10s |\n", id, name, specialization);
System.out.println("+------------+-----------------+------------+");
```
