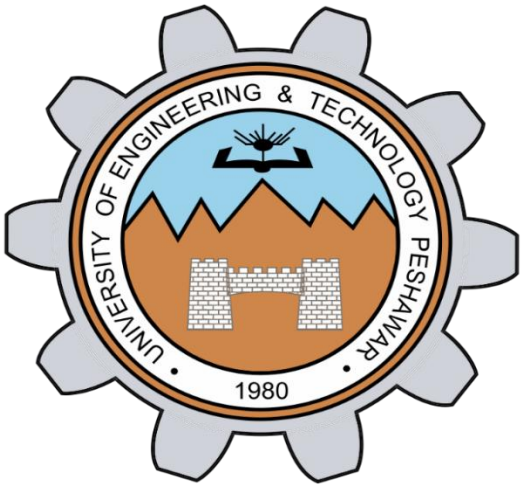
UNIVERSITY OF ENGINEERING AND TECHNOLOGY,
PESHAWAR PAKISTAN

Main Campus



Java Object Oriented Programming

FINAL PROJECT

Submitted By

Name: **Syed Sohaib Hassan**

Registration No: **F24CS499**

Semester: **2nd Semester, BS Data Science**

Section: **A**

Submitted To

SIR MUHAMMAD

DEPARTMENT OF COMPUTER SCIENCE

UNIVERSITY OF ENGINEERING AND TECHNOLOGY, PESHAWAR, PAKISTAN

Hostel Management System

1. Project Overview

The Hostel Management System is a comprehensive JavaFX application designed to manage various aspects of hostel operations. The system provides functionality for managing students, staff, rooms, allocations, payments, and maintenance reports through an intuitive graphical user interface.

2. System Architecture

2.1 Technology Stack

Programming Language: **Java**

UI Framework: **JavaFX**

UI Definition: **FXML**

Styling: **CSS**

Database Connectivity: **JDBC**

2.2 Application Structure

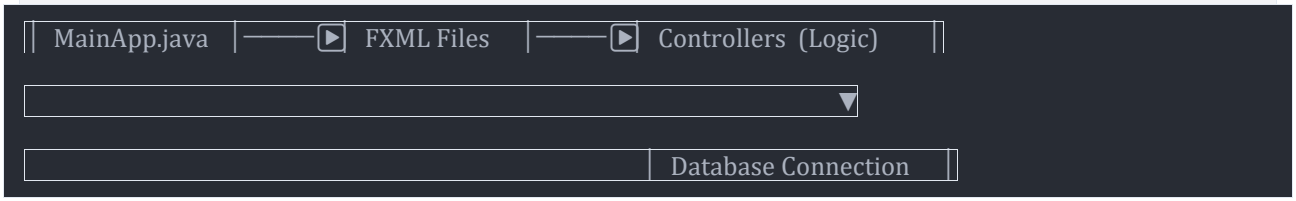
The application follows the Model-View-Controller (MVC) architectural pattern:

Model: **Database entities (Students, Staff, Rooms, etc.)**

View: **FXML files defining the UI layout**

Controller: **Java classes handling user interactions and business logic**

2.3 Component Diagram



3. Module Descriptions

3.1 Main Dashboard

The main dashboard serves as the central navigation hub of the application. It provides buttons to access all the management modules:

Student Management

Staff Management

Room Management

Allocation Management

Payment Management

Maintenance Reports

The dashboard is defined in `MainDashboard.fxml` and controlled by `MainDashboardController.java`.

3.2 Student Management

The Student Management module allows administrators to:

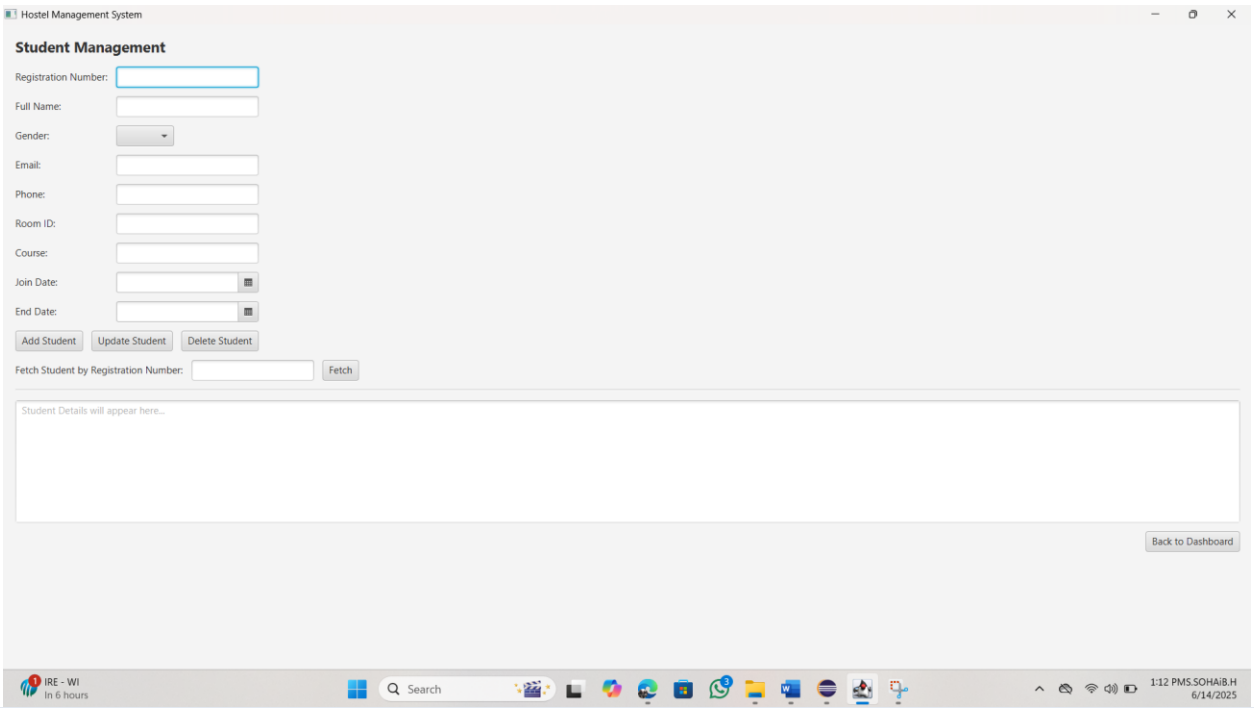
Add new students with details (registration number, name, gender, email, phone, room ID, course, join date, end date)

Update existing student information

Delete student records

Search for students by registration number

The module validates room IDs against the database to ensure they exist before assigning students.



3.3 Staff Management

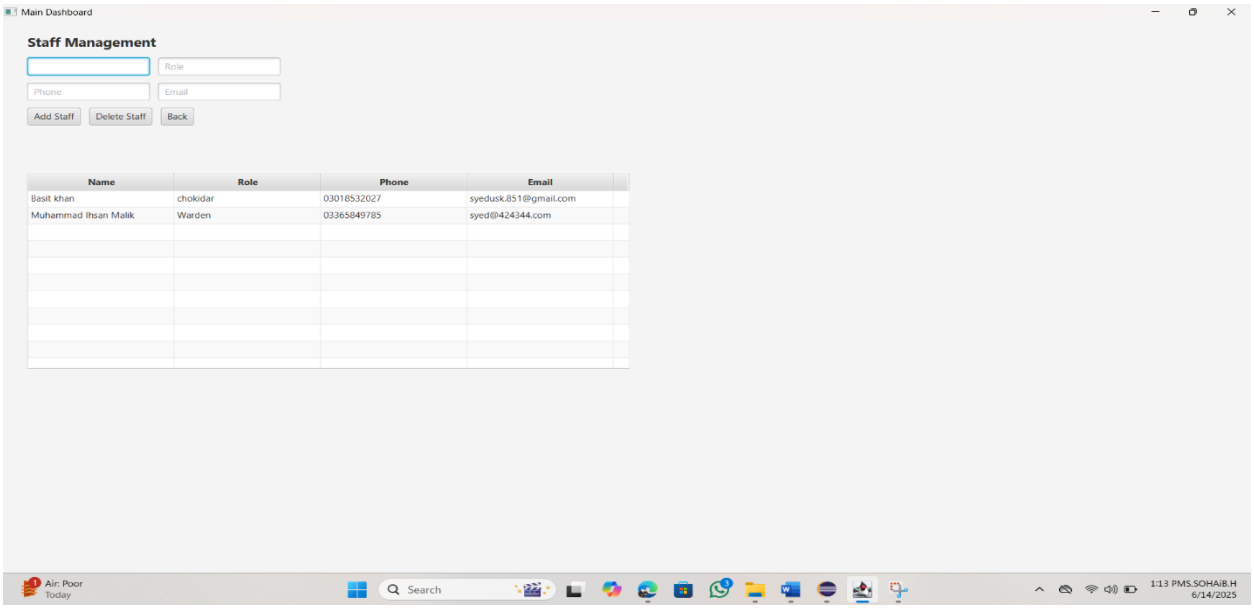
The Staff Management module provides functionality to:

Add new staff members with their details

Update staff information

Delete staff records

Search for staff by ID

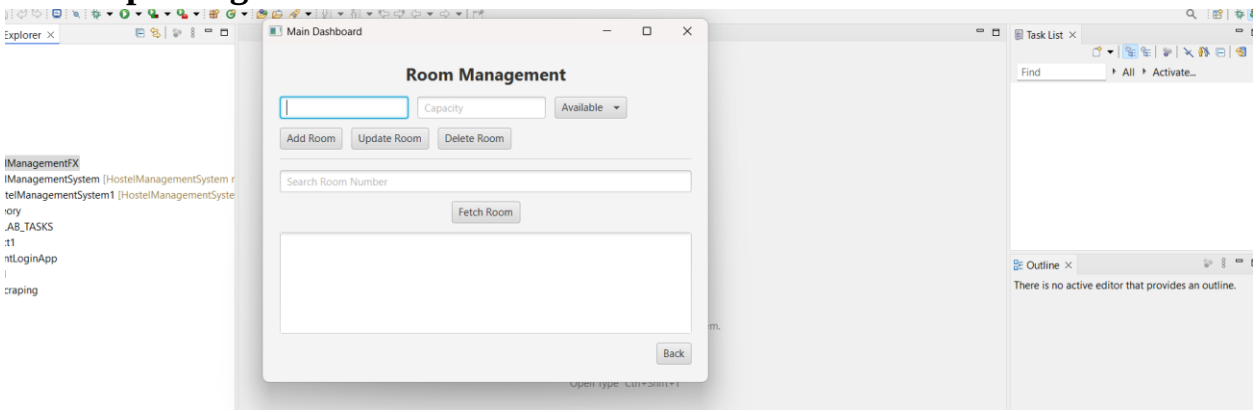


3.4 Room Management

The Room Management module enables:

Adding new rooms with room number and capacity

Updating room details



Deleting room records

Searching for rooms by room number

3.5 Allocation Management

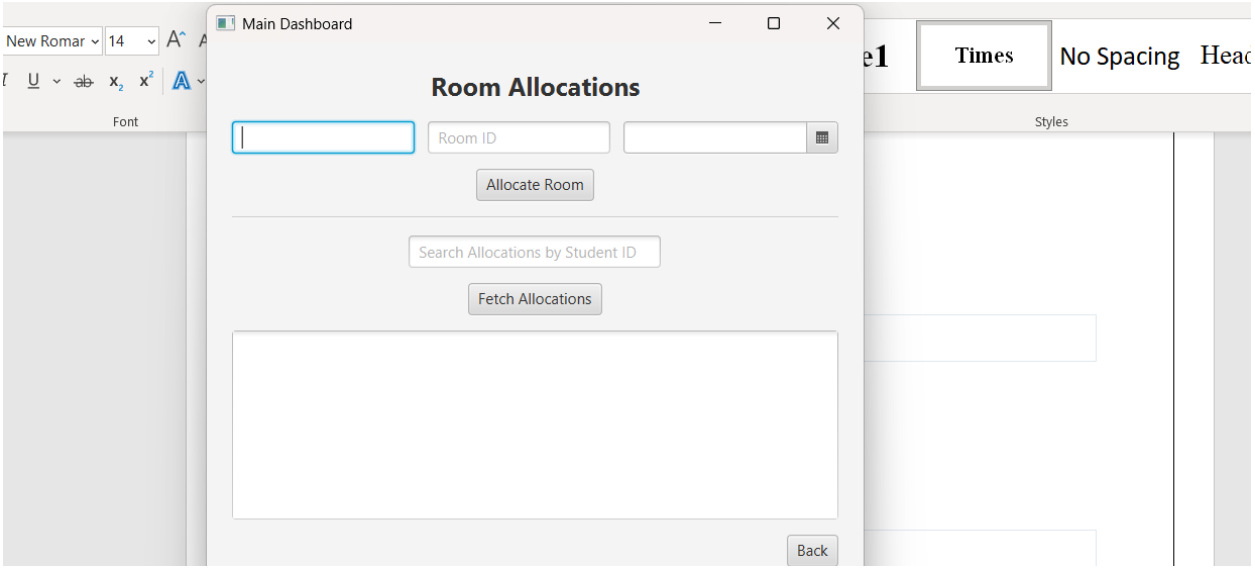
This module handles the allocation of rooms to students:

Assign students to specific rooms

Record allocation dates

Update or remove allocations

View allocation details



3.6 Payment Management

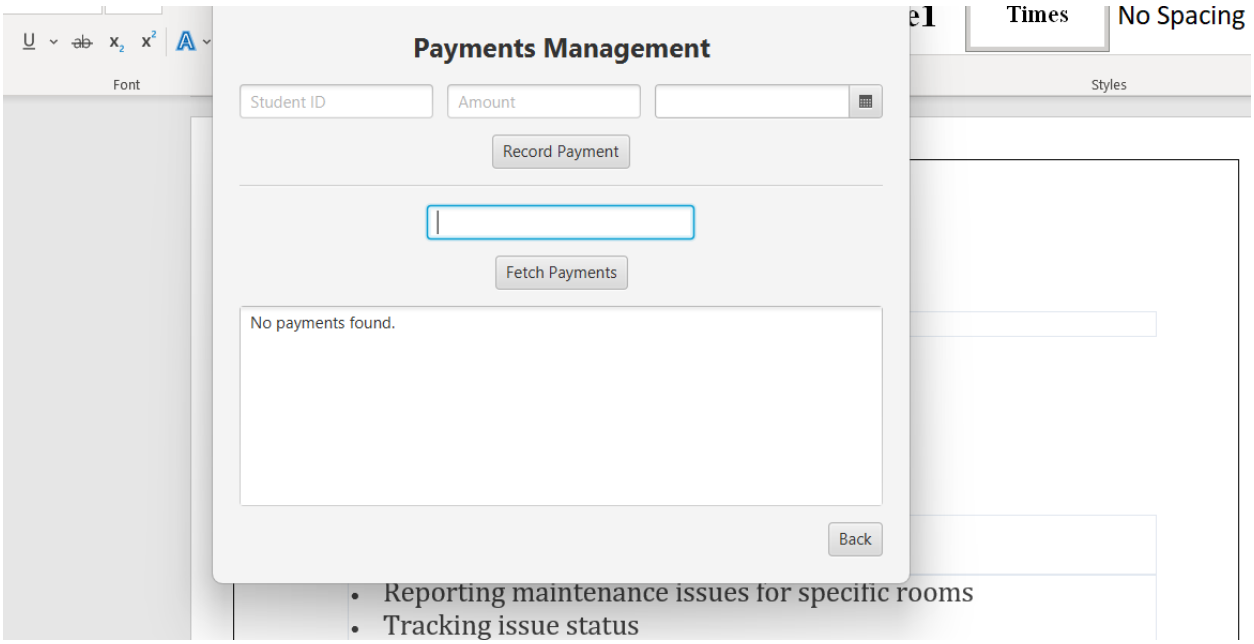
The Payment Management module tracks financial transactions:

Record payments made by students

Specify payment amounts and dates

Search payment history

Generate payment reports



3.7 Maintenance Management

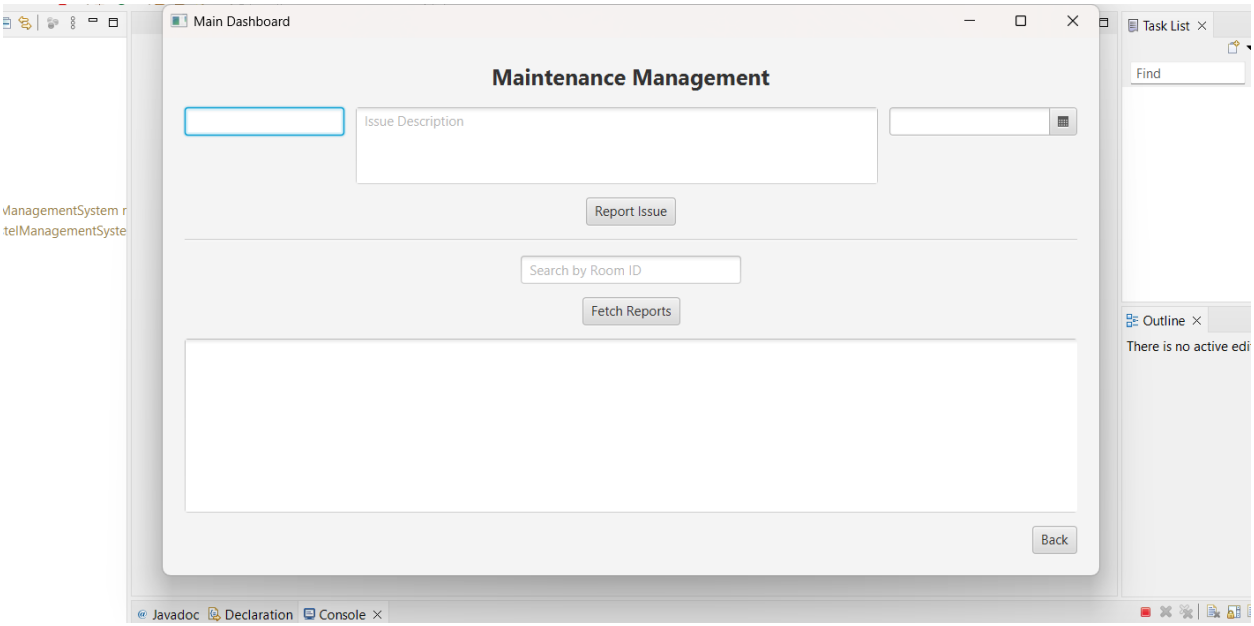
This module allows for:

Reporting maintenance issues for specific rooms

Tracking issue status

Recording resolution details

Viewing maintenance history



4. Database Structure

The application connects to a database with the following tables:

students

staff

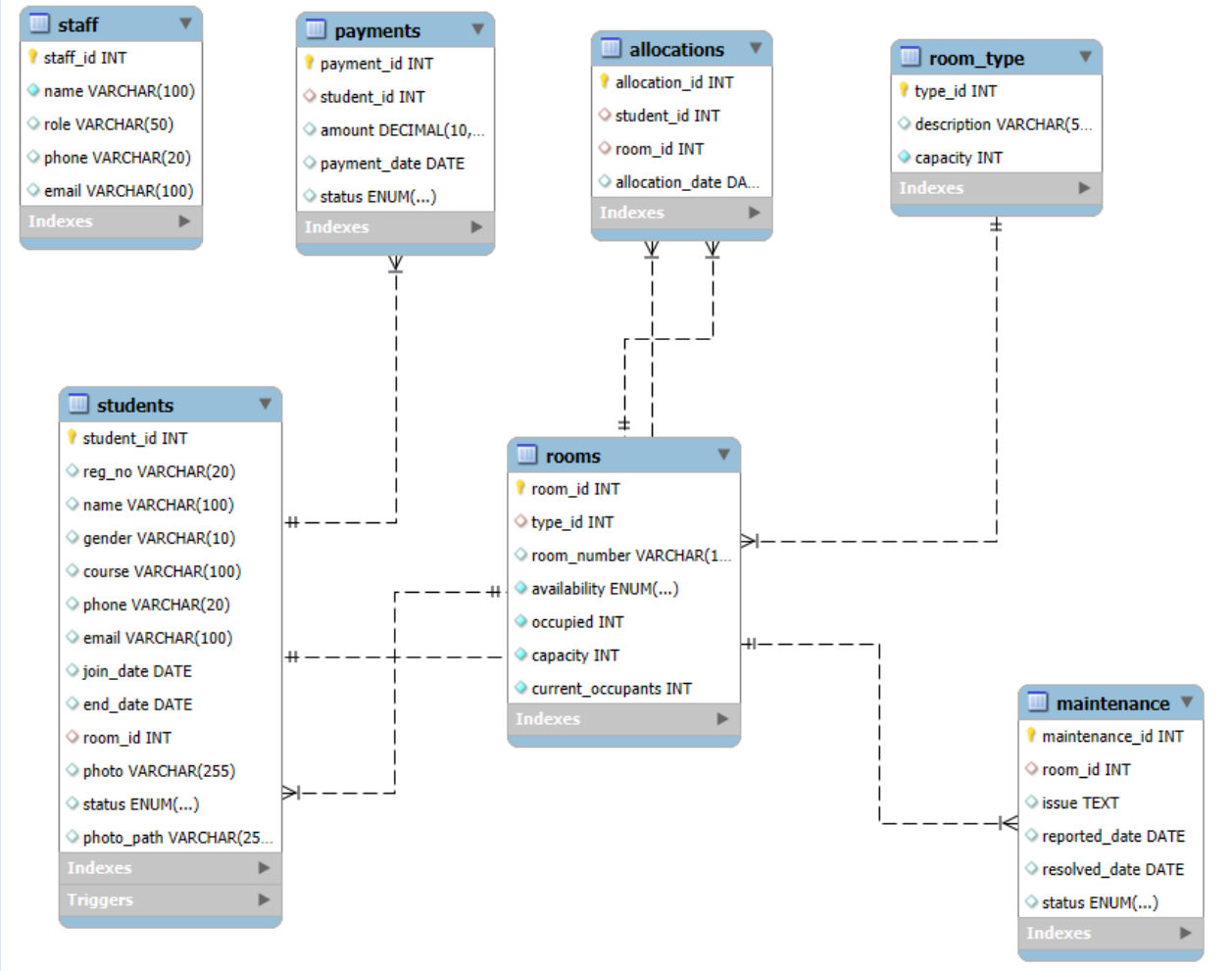
rooms

allocations

payments

maintenance_reports

The database connection is managed through the `DatabaseConnection` class, which provides methods for establishing and closing connections.



SQL CODE:

```
CREATE DATABASE IF NOT EXISTS hostel1_db1;
USE hostel1_db1;


CREATE TABLE room_type (
   type_id INT AUTO_INCREMENT PRIMARY KEY,
   description VARCHAR(100) NOT NULL
);

CREATE TABLE rooms (
   room_id INT AUTO_INCREMENT PRIMARY KEY,
   type_id INT,
   room_number VARCHAR(20) UNIQUE NOT NULL,
   availability ENUM('Available', 'Occupied') DEFAULT 'Available',
   capacity INT NOT NULL,
   current_occupants INT DEFAULT 0,
   FOREIGN KEY (type_id) REFERENCES room_type(type_id) ON DELETE SET NULL
);

CREATE TABLE students (
   student_id INT AUTO_INCREMENT PRIMARY KEY,
   reg_no VARCHAR(20) UNIQUE NOT NULL,
   name VARCHAR(100) NOT NULL,
   gender ENUM('Male', 'Female') NOT NULL,
   email VARCHAR(100),
   phone VARCHAR(20),
   course VARCHAR(100),
   join_date DATE,
   end_date DATE,
   room_id INT,
   photo_path VARCHAR(255),
   FOREIGN KEY (room_id) REFERENCES rooms(room_id) ON DELETE SET NULL
);
CREATE TABLE staff (
   staff_id INT AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(100) NOT NULL,
   role VARCHAR(100),
   phone VARCHAR(20),
   email VARCHAR(100)
);


CREATE TABLE maintenance
```

```
(
    maintenance_id INT AUTO_INCREMENT PRIMARY KEY,
    room_id INT NOT NULL,
    issue TEXT NOT NULL,
    reported_date DATE NOT NULL,
    resolved_date DATE,
    status ENUM('Pending', 'In Progress', 'Completed') DEFAULT 'Pending',
    FOREIGN KEY (room_id) REFERENCES rooms(room_id) ON DELETE CASCADE
);
```

-- 1. Students with room number:

```
    SELECT s.*, r.room_number FROM students s LEFT JOIN rooms r ON s.room_id =
r.room_id;
```

-- 2. Maintenance with room number:

```
  SELECT m.*, r.room_number FROM maintenance m JOIN rooms r ON m.room_id =
r.room_id;
```

-- 3. All available rooms:

```
    SELECT * FROM rooms WHERE availability = 'Available';
```

-- 4. Room occupancy check:

```
  SELECT room_number, capacity, current_occupants FROM rooms;
```

-- 5. All staff:

```
  SELECT * FROM staff;
```

-- 6. Specific student by reg_no:

```
SELECT s.*, r.room_number FROM students s LEFT JOIN rooms r ON s.room_id = r.room_id
WHERE s.reg_no = 'REG123';
```

-- 7. Get full room details including type:

```
  SELECT r.*, rt.description FROM rooms r LEFT JOIN room_type rt ON r.type_id = rt.type_id;
```

5. User Interface Design

The user interface is designed with a clean, modern look using JavaFX components styled with CSS. Key UI features include:

- **Consistent blue color scheme (#4285F4)**
- **Responsive buttons with hover effects**
- **Form validation with error alerts**
- **Navigation between modules**
- **Text fields with focus effects**
- **Clean typography using "Segoe UI" font family**

6. How to Run the Application

6.1 Prerequisites

- **Java Development Kit (JDK) 8 or higher**
- **JavaFX libraries (included in JDK 8-10, separate download for JDK 11+)**
- **MySQL or another compatible database server**
- **Maven**

6.2 Database Setup

- **Create a new database named** `hostel1_db1.`
- **Execute the SQL script to create the required tables (script not provided in the files)**
- **Update database connection parameters in the** `DatabaseConnection` **class if necessary**

6.3 Running the Application

if using an IDE like Eclipse or IntelliJ IDEA:

Import the project

Set up the project dependencies (JavaFX libraries)

Run the `MainApp` class

6.4 Testing Database Connection

You can test the database connection by running the **`DBTest`** class:

```
java -cp bin com.hostel1.DBTest
```

10. Conclusion

The Hostel Management System provides a comprehensive solution for managing hostel operations. Its modular design allows for easy maintenance and future enhancements. The intuitive user interface ensures that administrators can efficiently manage all aspects of hostel operations from a single application.

😊END 😊