

# Hostel Management Website

## Report

### Experiential Learning/Case Study/ Course Project

NAME	EMAIL	REGD. NO	CONTACT.NO	ROLL.NO
SHASANK KUMAR	<a href="mailto:2101020302@cgu-odisha.ac.in">2101020302@cgu-odisha.ac.in</a>	2101020302	6370146676	CIT21062
AUFIYA SUROOR	<a href="mailto:2101020303@cgu-odisha.ac.in">2101020303@cgu-odisha.ac.in</a>	2101020303	8117961496	CIT21063
ABHIPSA MAHAPATRA	<a href="mailto:2101020304@cgu-odisha.ac.in">2101020304@cgu-odisha.ac.in</a>	2101020304	9438206287	CIT21064
SAURAV KUMAR	<a href="mailto:2101020305@cgu-odisha.ac.in">2101020305@cgu-odisha.ac.in</a>	2101020305	7717707825	CIT21065
RANA SUJEET KUMAR	<a href="mailto:2101020306@cgu-odisha.ac.in">2101020306@cgu-odisha.ac.in</a>	2101020306	6203370346	CIT21066
SIBOM SAHU	<a href="mailto:2101020307@cgu-odisha.ac.in">2101020307@cgu-odisha.ac.in</a>	2101020307	8917684148	CIT21067

under the supervision of

Dr. Rakesh Ranjan kumar



Department of Computer Science and Engineering  
C.V. Raman Global University, Bhubaneswar  
Odisha, 752054, India

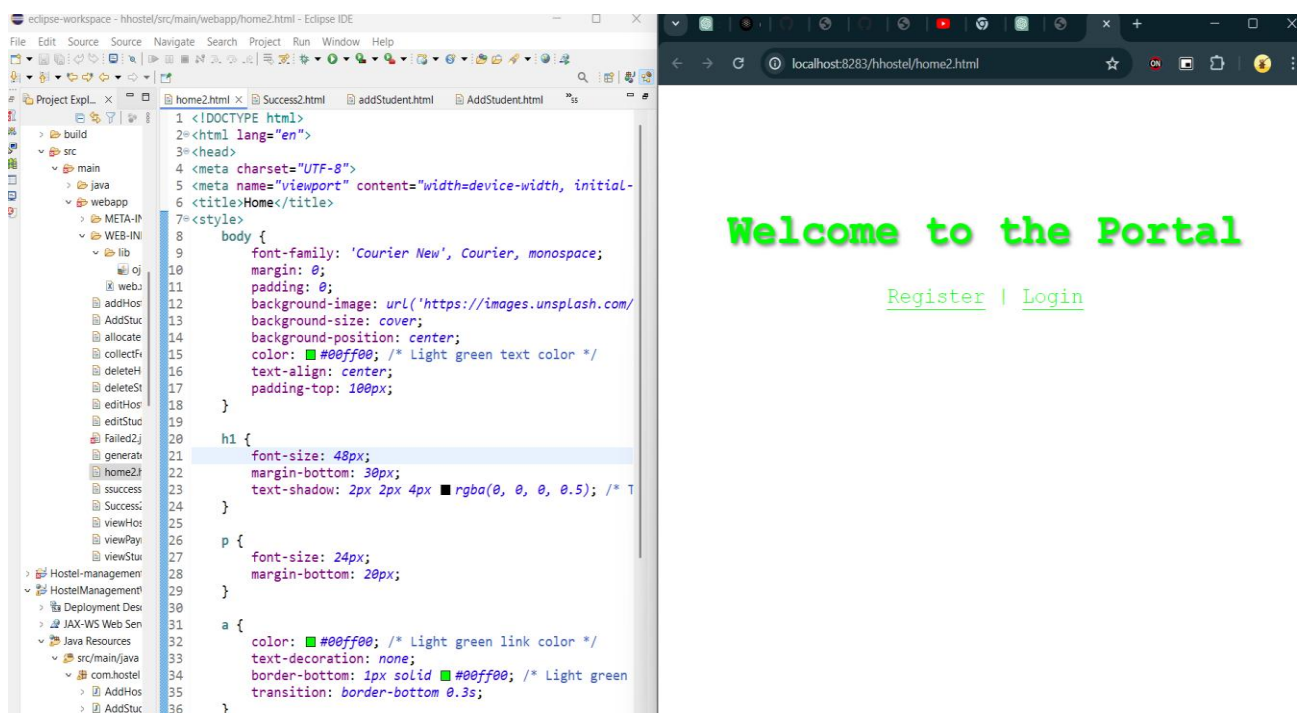
## **Abstract**

The Hostel Management System framework is a java website designed to offer facilities to both staff and students, hence reducing the time spent on paperwork. The number of dorm buildings is growing due to the number of students who are willing to live there while they study, thus it is important to manage them wisely by employing online applications that relieve pressure on the authorities. To access the application dashboard, which allows them to check student records and make updates as needed, administrators and students must provide their login credentials. From there, they can all readily access information about their registration for hostel rooms, fee payments, and other related matters. The shortcomings of the previous management approaches are addressed by this application.

## Introduction

Managing a hostel is a demanding task that requires meticulous supervision and significant time investment. In such circumstances, it becomes imperative for hostels to employ software solutions that streamline hostel management processes. Traditional paper-based methods are not only labor-intensive but also prone to errors, leading to suboptimal resource utilization and potentially negative impacts on both the hostel and the institution it serves.

This project aims to address these challenges by providing a comprehensive software solution tailored to the needs of both students and administration groups. By digitizing record-keeping processes, it encourages efficient management of hostel resources. The project leverages the capabilities of Java programming language and SQL for database management, offering a robust and scalable solution.



## Technologies Used:

- Development Environment: Eclipse IDE
- Database Management System: MySQL (utilizing XAMPP)
- Server: Apache Tomcat

- Programming Languages: Java (for backend logic) and SQL (for database operations)

### **Key Features:**

- User-friendly Interface: Utilizing Eclipse's GUI builder for intuitive layout design and drag-and-drop interface components.
- Secure Authentication: Students are provided with unique login IDs (primary keys) and passwords for accessing their information securely.
- Dynamic Student Profiles: The system allows students to view and update their personal information as needed.
- Administrative Control: Wardens and administrators have access to all hostel-related functions, including managing student records and accommodation allocations.
- Streamlined Registration: The system facilitates a smooth registration process, reducing administrative workload and paperwork while enhancing the overall registration cycle efficiency.

## **2 Related work**

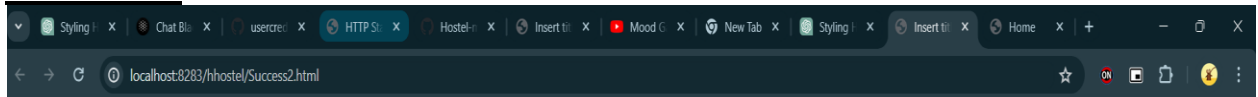
There are numerous comparable hostel management systems available, each with a unique feature set and strategy for addressing the difficulties involved in managing a hostel. Among the noteworthy systems are:

Hospital Administration Platform from XYZ Technologies: Features like online room assignment, fee payment, and attendance tracking are available with this system. It improves overall usefulness by offering an interface that is easy to use for administrators and students alike.

HostelPro: With features like inventory management, visitor tracking, and food management, HostelPro is a complete hostel management solution. It is renowned for being durable and scalable, accommodating hostels of all sizes.

HostelMate: This hostel management system, which focuses on automation and integration, allows for smooth communication between various modules, including room assignment, money collection, and maintenance tracking. Because of its integration possibilities, it is a recommended option.

## Dashboard-



### Welcome to Hostel Management System

[Add Hostel](#)  
[Edit Hostel](#)  
[Delete Hostel](#)  
[View Hostel Details](#)  
[Add Student](#)  
[Edit Student](#)  
[Delete Student](#)  
[Allocate Room](#)  
[View Student Details](#)  
[Collect Fee](#)  
[View Payment History](#)  
[Generate Reports](#)

## 3 Proposed Model

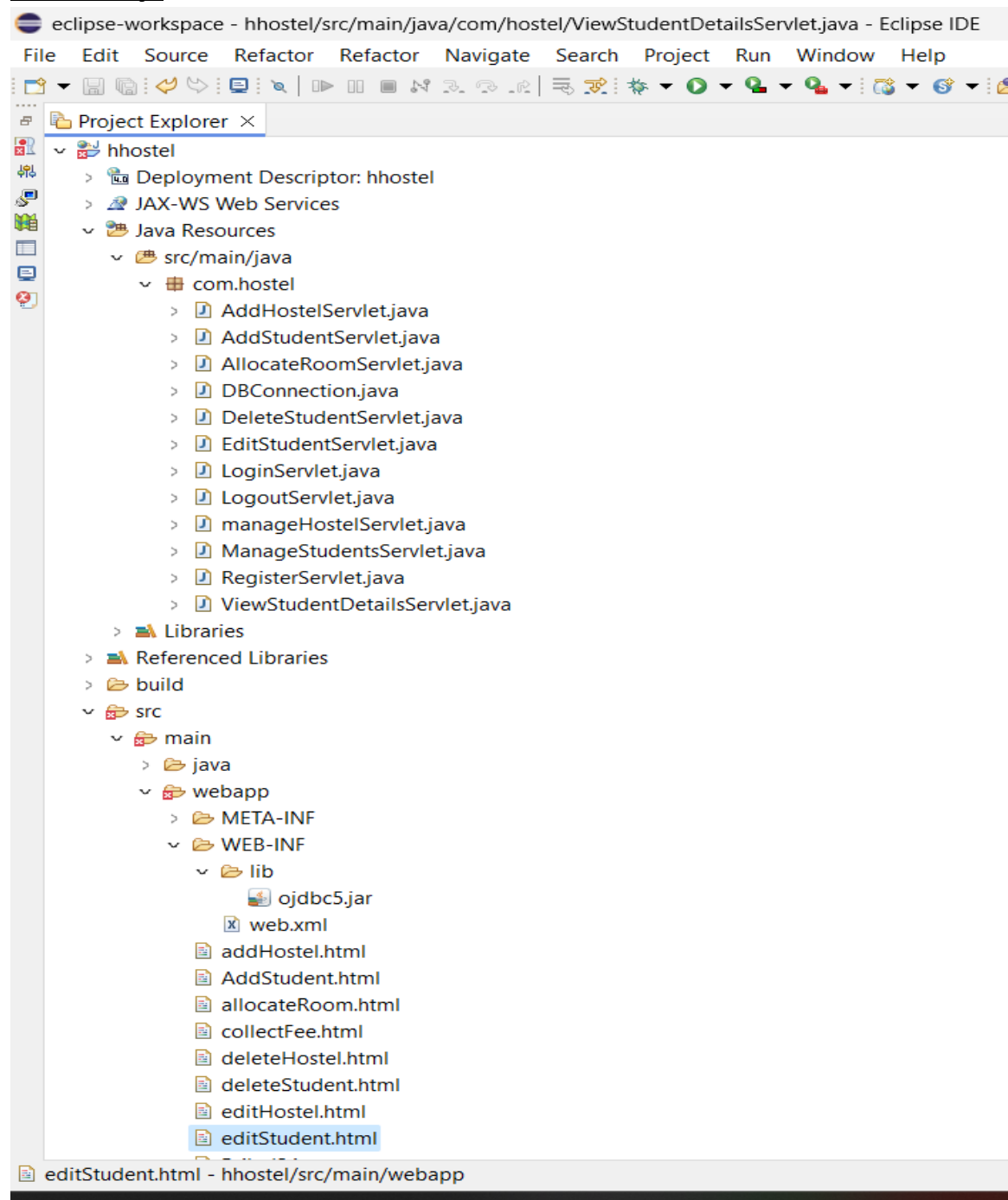
The proposed work intends to significantly increase hostel administration efficiency by introducing many enhancements and new features, building upon the existing hostel management systems. Among them are:

**Integration with Student Information System (SIS):** By integrating the residence hall management system with the school's SIS, student data, such as enrollment status, course specifics, and personal data, can flow seamlessly between the two systems, eliminating duplication of information and guaranteeing data consistency.

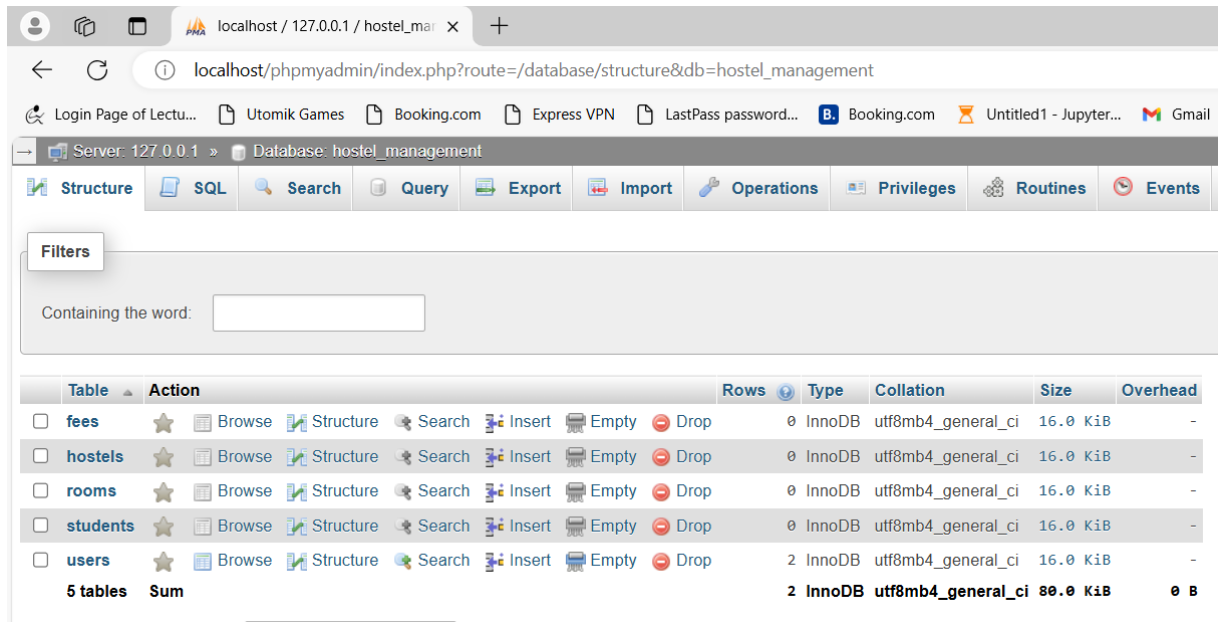
**Mobile Application Support:** By creating a companion mobile application for the web-based hostel administration system, users would have more accessibility and be able to utilize their cellphones for activities like reserving rooms, paying fees, and contacting the hostel administration.

**Analytics and Reporting Tools:** Administrators can gain insight into hostel occupancy by implementing analytics and reporting tools within the system.

## Directory-



## Database and table



The screenshot shows the phpMyAdmin interface for a database named 'hostel\_management'. The top navigation bar includes tabs for Structure, SQL, Search, Query, Export, Import, Operations, Privileges, Routines, and Events. Below the navigation bar is a 'Filters' section with a text input field labeled 'Containing the word:'. The main content area displays a table of database structures with columns: Table, Action, Rows, Type, Collation, Size, and Overhead. The table lists five tables: fees, hostels, rooms, students, and users. Each table has a set of actions (Browse, Structure, Search, Insert, Empty, Drop) and a star icon. The 'users' table is highlighted in blue. At the bottom, a summary row shows '5 tables' and a 'Sum' of rows and size.

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> fees	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> hostels	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> rooms	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> students	★ Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> users	★ Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB	-
5 tables	Sum	2	InnoDB	utf8mb4_general_ci	80.0 KiB	0 B

## Code

### AddHostelServlet

```
package com.hostel;
```

```
import java.io.IOException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
```

```
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
```

```
@WebServlet("/AddHostelServlet")
public class AddHostelServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;
```

```
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        String name = request.getParameter("name");
```

```

String location = request.getParameter("location");

try {
    Connection con = DBConnection.getConnection();
    PreparedStatement pst = con.prepareStatement("INSERT INTO hostels (name,
location) VALUES (?, ?)");
    pst.setString(1, name);
    pst.setString(2, location);
    pst.executeUpdate();
    con.close();
    // Redirect to a success page
    response.sendRedirect("success.html");
} catch (SQLException e) {
    e.printStackTrace();
    // Redirect to an error page
    response.sendRedirect("error.html");
}
}
} AddStudentServlet
package com.hostel;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;

import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;

@WebServlet("/addStudent")
public class AddStudentServlet extends HttpServlet {
    /**
     *
     */
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        String name = request.getParameter("name");
        String contactInfo = request.getParameter("contactInfo");

```



```

String course = request.getParameter("course");
int hostelID = Integer.parseInt(request.getParameter("hostelID"));
int roomID = Integer.parseInt(request.getParameter("roomID"));

try {
    Connection connection = DBConnection.getConnection();
    PreparedStatement preparedStatement = connection.prepareStatement("INSERT INTO
students (name, contactInfo, course, hostelID, roomID) VALUES (?, ?, ?, ?, ?)");
    preparedStatement.setString(1, name);
    preparedStatement.setString(2, contactInfo);
    preparedStatement.setString(3, course);
    preparedStatement.setInt(4, hostelID);
    preparedStatement.setInt(5, roomID);

    int rowsAffected = preparedStatement.executeUpdate();
    if (rowsAffected > 0) {
        out.println("Student added successfully!");
    } else {
        out.println("Failed to add student!");
    }

    connection.close();
} catch (SQLException e) {
    out.println("SQL Exception: " + e.getMessage());
}
out.close();
}
}

```

### AllocateRoomServlet

```

package com.hostel;

import jakarta.servlet.ServletException;

import jakarta.servlet.annotation.WebServlet;

import jakarta.servlet.http.HttpServlet;

import jakarta.servlet.http.HttpServletRequest;

import jakarta.servlet.http.HttpServletResponse;

import java.io.IOException;

import java.io.PrintWriter;

import java.sql.Connection;

import java.sql.PreparedStatement;

```

```
import java.sql.SQLException;

@WebServlet("/allocateRoom")

public class AllocateRoomServlet extends HttpServlet {

    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response)

        throws ServletException, IOException {

        response.setContentType("text/html");

        PrintWriter out = response.getWriter();

        int studentID = Integer.parseInt(request.getParameter("studentID"));

        int roomID = Integer.parseInt(request.getParameter("roomID"));

        try {

            Connection connection = DBConnection.getConnection();

            PreparedStatement preparedStatement = connection.prepareStatement("UPDATE
students SET roomID=? WHERE studentID=?");

            preparedStatement.setInt(1, roomID);

            preparedStatement.setInt(2, studentID);

            int rowsAffected = preparedStatement.executeUpdate();

            if (rowsAffected > 0) {

                out.println("Room allocated successfully!");

            } else {

                out.println("Failed to allocate room!");

            }

            connection.close();

        } catch (SQLException e) {

            out.println("SQL Exception: " + e.getMessage());

        }

    }

}
```

```
out.close();
```

```
}
```

```
}
```

## DBConnection

```
package com.hostel;
```

```
import java.sql.Connection;
```

```
import java.sql.DriverManager;
```

```
import java.sql.SQLException;
```

```
public class DBConnection {
```

```
    private static Connection con;
```

```
    private static String driver;
```

```
    private static String url;
```

```
    static {
```

```
        con = null;
```

```
        driver = "com.mysql.cj.jdbc.Driver";
```

```
        url = "jdbc:mysql://localhost:3306/hostel_management";
```

```
    }
```

```
    public DBConnection() {}
```

```
    public static Connection getConnection() {
```

```
        try {
```

```
            Class.forName(driver);
```

```
            con = DriverManager.getConnection(url, "root", "");
```

```
            return con;
```

```
        } catch (ClassNotFoundException | SQLException e) {
```

```
            System.out.println("Error: " + e.getMessage());
```

```
            return null;
```

```
        }
```

```
    }
```

```
    public static void main(String[] args) {
```

```
        Connection connection = null;
```

```
        try {
```

```
            connection = DBConnection.getConnection();
```

```
            if (connection != null) {
```

```
                System.out.println("Connection established successfully!");
```

```
                connection.close();
```

```
            } else {
```

```
                System.out.println("Failed to establish connection!");
```

```
            }
```

```
        } catch (SQLException e) {
```

```
            System.out.println("Error: " + e.getMessage());
```

```
        }
```

```
}  
}
```

### DeleteStudentServlet

```
package com.hostel;
```

```
import jakarta.servlet.ServletException;
```

```
import jakarta.servlet.annotation.WebServlet;
```

```
import jakarta.servlet.http.HttpServlet;
```

```
import jakarta.servlet.http.HttpServletRequest;
```

```
import jakarta.servlet.http.HttpServletResponse;
```

```
import java.io.IOException;
```

```
import java.io.PrintWriter;
```

```
import java.sql.Connection;
```

```
import java.sql.PreparedStatement;
```

```
import java.sql.SQLException;
```

```
@WebServlet("/deleteStudent")
```

```
public class DeleteStudentServlet extends HttpServlet {
```

```
    /**
```

```
     *
```

```
    */
```

```
    private static final long serialVersionUID = 1L;
```

```
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
```

```
        throws ServletException, IOException {
```

```
        response.setContentType("text/html");
```

```
        PrintWriter out = response.getWriter();
```

```
        int studentID = Integer.parseInt(request.getParameter("studentID"));
```

```
        try {
```

```

        Connection connection = DBConnection.getConnection();

        PreparedStatement preparedStatement = connection.prepareStatement("DELETE
FROM students WHERE studentID=?");

        preparedStatement.setInt(1, studentID);

        int rowsAffected = preparedStatement.executeUpdate();

        if (rowsAffected > 0) {

            out.println("Student deleted successfully!");

        } else {

            out.println("Failed to delete student!");

        }

        connection.close();

    } catch (SQLException e) {

        out.println("SQL Exception: " + e.getMessage());

    }

    out.close();

}

}

EditStudentServlet
package com.hostel;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.SQLException;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;

@WebServlet("/editStudent")
public class EditStudentServlet extends HttpServlet {
    /** * */
    private static final long serialVersionUID = 1L;
    protected void doPost(HttpServletRequest request, HttpServletResponse response)

```

```

        throws ServletException, IOException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();

    int studentID = Integer.parseInt(request.getParameter("studentID"));
    String name = request.getParameter("name");
    String contactInfo = request.getParameter("contactInfo");
    String course = request.getParameter("course");
    int hostelID = Integer.parseInt(request.getParameter("hostelID"));
    int roomID = Integer.parseInt(request.getParameter("roomID"));

    try {
        Connection connection = DBConnection.getConnection();
        PreparedStatement preparedStatement = connection.prepareStatement("UPDATE
students SET name=?, contactInfo=?, course=?, hostelID=?, roomID=? WHERE
studentID=?");
        preparedStatement.setString(1, name);
        preparedStatement.setString(2, contactInfo);
        preparedStatement.setString(3, course);
        preparedStatement.setInt(4, hostelID);
        preparedStatement.setInt(5, roomID);
        preparedStatement.setInt(6, studentID);

        int rowsAffected = preparedStatement.executeUpdate();
        if (rowsAffected > 0) {
            out.println("Student updated successfully!");
        } else {
            out.println("Failed to update student!");
        }
        connection.close();
    } catch (SQLException e) {
        out.println("SQL Exception: " + e.getMessage());
    }
    out.close();
}
}

```

### LoginServlet

```

package com.hostel;
import java.io.IOException;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import jakarta.servlet.http.HttpSession;
@WebServlet("/LoginServlet")
public class LoginServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

```

```

protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    String username = request.getParameter("username");
    String password = request.getParameter("password");
    try {
        Connection con = DBConnection.getConnection();
        PreparedStatement pst = con.prepareStatement("select * from users where username=?
and password=?");
        pst.setString(1, username);
        pst.setString(2, password);
        ResultSet rs = pst.executeQuery();
        if (rs.next()) {
            HttpSession session = request.getSession();
            session.setAttribute("username", username);
            // Redirect to dashboard if login successful
            response.sendRedirect("dashboard.jsp");
        } else {
            // Redirect back to login page with error message
            response.sendRedirect("login.jsp?error=1");
        }
        con.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

```

### ViewStudentDetailsServlet

```

package com.hostel;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
@WebServlet("/viewStudentDetails")
public class ViewStudentDetailsServlet extends HttpServlet {
    /** */
    private static final long serialVersionUID = 1L;
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        int studentID = Integer.parseInt(request.getParameter("studentID"));

```

```

try {
    Connection connection = DBConnection.getConnection();
    PreparedStatement preparedStatement = connection.prepareStatement("SELECT *
FROM students WHERE studentID=?");
    preparedStatement.setInt(1, studentID);
    ResultSet resultSet = preparedStatement.executeQuery();
    if (resultSet.next()) {
        out.println("<h2>Student Details</h2>");
        out.println("<p>Name: " + resultSet.getString("name") + "</p>");
        out.println("<p>Contact Info: " + resultSet.getString("contactInfo") + "</p>");
        out.println("<p>Course: " + resultSet.getString("course") + "</p>");
        out.println("<p>Hostel ID: " + resultSet.getInt("hostelID") + "</p>");
        out.println("<p>Room ID: " + resultSet.getInt("roomID") + "</p>");
    } else {
        out.println("Student not found!");
    }
    connection.close();
} catch (SQLException e) {
    out.println("SQL Exception: " + e.getMessage());
}
out.close()}}

```

#### addHostel.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Add Hostel</title>
</head>
<body>
    <h2>Add Hostel</h2>
    <form action="addHostel" method="post">
        <label for="hostelName">Hostel Name:</label><br>
        <input type="text" id="hostelName" name="hostelName" required><br>
        <label for="address">Address:</label><br>
        <input type="text" id="address" name="address" required><br>
        <label for="capacity">Capacity:</label><br>
        <input type="number" id="capacity" name="capacity" required><br>
        <label for="facilities">Facilities:</label><br>
        <textarea id="facilities" name="facilities" rows="4" cols="50"></textarea><br><br>
        <input type="submit" value="Add Hostel">
    </form>
</body>
</html>

```

#### addStudent.html

```

<!DOCTYPE html>
<html lang="en">
<head>

```



```

<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Add Student</title>
</head>
<body>
<h2>Add Student</h2>
<form action="addStudent" method="post">
  <label for="name">Name:</label><br>
  <input type="text" id="name" name="name" required><br>
  <label for="studentID">studentID:</label><br>
  <input type="number" id="studentID" name="studentID" required><br><br>

  <label for="contactInfo">Contact Info:</label><br>
  <input type="text" id="contactInfo" name="contactInfo" required><br>
  <label for="course">Course:</label><br>
  <input type="text" id="course" name="course" required><br>
  <label for="hostelID">Hostel ID:</label><br>
  <input type="number" id="hostelID" name="hostelID" required><br>
  <label for="roomID">Room ID:</label><br>
  <input type="number" id="roomID" name="roomID" required><br><br>
  <input type="submit" value="Add Student">
</form>
</body>
</html>

```

### allocateRoom.html

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Allocate Room</title>
</head>
<body>
<h2>Allocate Room</h2>
<form action="allocateRoom" method="post">
  <label for="studentID">Student ID:</label><br>
  <input type="number" id="studentID" name="studentID" required><br>
  <label for="roomID">Room ID:</label><br>
  <input type="number" id="roomID" name="roomID" required><br><br>
  <input type="submit" value="Allocate Room">
</form>
</body>
</html>

```

### collectFee.html

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Collect Fee</title>

```

```

</head>
<body>
  <h2>Collect Fee</h2>
  <form action="collectFee" method="post">
    <label for="studentID">Student ID:</label><br>
    <input type="number" id="studentID" name="studentID" required><br>
    <label for="amount">Amount:</label><br>
    <input type="number" id="amount" name="amount" step="0.01" required><br><br>
    <input type="submit" value="Collect Fee">
  </form>
</body>
</html>
deleteHostel.html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Delete Hostel</title>
</head>
<body>
  <h2>Delete Hostel</h2>
  <form action="" method="post">
    <label for="hostelID">Hostel ID:</label><br>
    <input type="number" id="hosteldeleteHostelID" name="hostelID" required><br><br>
    <input type="submit" value="Delete Hostel">
  </form>
</body>
</html>
deleteStudent.html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Delete Student</title>
</head>
<body>
  <h2>Delete Student</h2>
  <form action="deleteStudent" method="post">
    <label for="studentID">Student ID:</label><br>
    <input type="number" id="studentID" name="studentID" required><br><br>
    <input type="submit" value="Delete Student">
  </form>
</body>
</html>
editHostel.html
<!DOCTYPE html>
<html lang="en">
<head>

```

```

<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Edit Hostel</title>
</head>
<body>
<h2>Edit Hostel</h2>
<form action="editHostel" method="post">
  <label for="hostelID">Hostel ID:</label><br>
  <input type="number" id="hostelID" name="hostelID" required><br>
  <label for="hostelName">Hostel Name:</label><br>
  <input type="text" id="hostelName" name="hostelName" required><br>
  <label for="address">Address:</label><br>
  <input type="text" id="address" name="address" required><br>
  <label for="capacity">Capacity:</label><br>
  <input type="number" id="capacity" name="capacity" required><br>
  <label for="facilities">Facilities:</label><br>
  <textarea id="facilities" name="facilities" rows="4" cols="50"></textarea><br><br>
  <input type="submit" value="Edit Hostel">
</form>
</body>
</html>

```

**editStudent.html**

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Edit Student</title>
</head>
<body>
  <h2>Edit Student</h2>
  <form action="editStudent" method="post">
    <label for="studentID">Student ID:</label><br>
    <input type="number" id="studentID" name="studentID" required><br>
    <label for="name">Name:</label><br>
    <input type="text" id="name" name="name" required><br>
    <label for="contactInfo">Contact Info:</label><br>
    <input type="text" id="contactInfo" name="contactInfo" required><br>
    <label for="course">Course:</label><br>
    <input type="text" id="course" name="course" required><br>
    <label for="hostelID">Hostel ID:</label><br>
    <input type="number" id="hostelID" name="hostelID" required><br>
    <label for="roomID">Room ID:</label><br>
    <input type="number" id="roomID" name="roomID" required><br><br>
    <input type="submit" value="Edit Student">
  </form>
</body>
</html>

```

**Failed2.html**

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Failed</title>
    <style>
        body {
            font-family: 'Courier New', Courier, monospace;
            margin: 0;
            padding: 0;
            background-image: url('https://images.unsplash.com/photo-
1588441689852-9b402ff9c3a0'); /* Hacker/cybersecurity background
image */
            background-size: cover;
            background-position: center;
            color: #ff0000; /* Red text color */
            text-align: center;
            padding-top: 100px;
        }

        h1 {
            font-size: 48px;
            margin-bottom: 30px;
            text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5); /* Text
shadow for better readability */
        }

        p {
            font-size: 24px;
            margin-bottom: 20px;
        }

        a {
            color: #00ff00; /* Light green link color */
            text-decoration: none;
            border-bottom: 1px solid #00ff00; /* Light green
underline */
            transition: border-bottom 0.3s;

            a:hover {
                border-bottom: 2px solid #00ff00; /* Light green thicker
underline on hover */
            }
        }
    </style>
```

```

</head>
<body>
  <h1>Failed!</h1>
  <p>Registration/login failed. Please try again.</p>
  <p><a href="home2.html">Go to Home</a></p>
</body>
</html>

```

### generateReports.html

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Generate Reports</title>
</head>
<body>
  <h2>Generate Reports</h2>
  <form action="generateReports" method="post">
    <!-- Add options for selecting report type, date range, etc.
-->

    <!-- Example: -->
    <label for="reportType">Report Type:</label><br>
    <select id="reportType" name="reportType">
      <option value="occupancy">Occupancy Report</option>
      <option value="feeCollection">Fee Collection
Report</option>
      <!-- Add more options as needed -->
    </select><br><br>
    <input type="submit" value="Generate Report">
  </form>
</body>
</html>

```

### home2.html

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Home</title>
<style>
  body {
    font-family: 'Courier New', Courier, monospace;
    margin: 0;
    padding: 0;

```

```

        background-image: url('https://images.unsplash.com/photo-
1588441689852-9b402ff9c3a0'); /* Hacker/cybersecurity background
image */
        background-size: cover;
        background-position: center;
        color: #00ff00; /* Light green text color */
        text-align: center;
        padding-top: 100px;
    }

    h1 {
        font-size: 48px;
        margin-bottom: 30px;
        text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5); /* Text shadow
for better readability */
    }

    p {
        font-size: 24px;
        margin-bottom: 20px;
    }

    a {
        color: #00ff00; /* Light green link color */
        text-decoration: none;
        border-bottom: 1px solid #00ff00; /* Light green underline */
        transition: border-bottom 0.3s;
    }

    a:hover {
        border-bottom: 2px solid #00ff00; /* Light green thicker
underline on hover */
    }
</style>
</head>
<body>
    <h1>Welcome to the Portal</h1>
    <p><a href="register2.html">Register</a> | <a
href="login2.html">Login</a></p>
</body>
</html>
success2.html
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Success</title>
    <style>
        body {

```

```

        font-family: 'Courier New', Courier, monospace;
        margin: 0;
        padding: 0;
        background-image: url('https://images.unsplash.com/photo-
1588441689852-9b402ff9c3a0');
        background-size: cover;
        background-position: center;
        color: #00ff00;
        text-align: center;
        padding-top: 100px;
    }

    h2 {
        font-size: 36px;
        margin-bottom: 20px;
        text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);
    }

    p {
        font-size: 24px;
        margin-bottom: 20px;
        color: #00ff00;
    }

    a {
        color: #00ff00;
        text-decoration: none;
        border-bottom: 1px solid #00ff00;
        transition: border-bottom 0.3s;
    }

    a:hover {
        border-bottom: 2px solid #00ff00;
    }
</style>
</head>
<body>
    <h2>Success</h2>
    <p>Your transaction was successful!</p>
    <a href="Success2.html">Go back to Home</a>
</body>
</html>
Success2.html
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
<style>

```

```

body {
    font-family: 'Courier New', Courier, monospace;
    margin: 0;
    padding: 0;
    background-image: url('https://images.unsplash.com/photo-1588441689852-9b402ff9c3a0'); /* Hacker/cybersecurity background image */
    background-size: cover;
    background-position: center;
    color: #00ff00; /* Light green text color */
    text-align: center;
    padding-top: 100px;
}

h1 {
    color: blue; /* Blue header color */
}

h2 {
    color: red; /* Red subheader color */
}

ul {
    list-style-type: none;
    padding: 0;
}

li {
    margin-bottom: 10px;
}

a {
    color: #00ff00; /* Light green link color */
    text-decoration: none;
    border-bottom: 1px solid #00ff00; /* Light green underline */
    transition: border-bottom 0.3s;
}

a:hover {
    border-bottom: 2px solid #00ff00; /* Light green thicker underline on hover */
}
</style>
</head>
<h1>Welcome to Hostel Management System</h1>
<ul>
    <li><a href="addHostel.html">Add Hostel</a></li>
    <li><a href="editHostel.html">Edit Hostel</a></li>
    <li><a href="deleteHostel.html">Delete Hostel</a></li>

```



```

        <li><a href="viewHostelDetails.html">View Hostel
Details</a></li>
        <li><a href="addStudent.html">Add Student</a></li>
        <li><a href="editStudent.html">Edit Student</a></li>
        <li><a href="deleteStudent.html">Delete Student</a></li>
        <li><a href="allocateRoom.html">Allocate Room</a></li>
        <li><a href="viewStudentDetails.html">View Student
Details</a></li>
        <li><a href="collectFee.html">Collect Fee</a></li>
        <li><a href="viewPaymentHistory.html">View Payment
History</a></li>
        <li><a href="generateReports.html">Generate Reports</a></li>
    </ul>
</body>
</html>

```

#### viewHostelDetails.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>View Hostel Details</title>
</head>
<body>
    <h2>View Hostel Details</h2>
    <form action="viewHostelDetails" method="post">
        <label for="hostelID">Hostel ID:</label><br>
        <input type="number" id="hostelID" name="hostelID"
required><br><br>
        <input type="submit" value="View Details">
    </form>
</body>
</html>

```

#### viewPaymentHistory.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>View Payment History</title>
</head>
<body>
    <h2>View Payment History</h2>
    <form action="viewPaymentHistory" method="post">
        <label for="studentID">Student ID:</label><br>
        <input type="number" id="studentID" name="studentID"
required><br><br>

```

```

        <input type="submit" value="View Payment History">
    </form>
</body>
</html>
viewStudentDetails.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>View Student Details</title>
</head>
<body>
    <h2>View Student Details</h2>
    <form action="viewStudentDetails" method="post">
        <label for="studentID">Student ID:</label><br>
        <input type="number" id="studentID" name="studentID"
required><br><br>
        <input type="submit" value="View Student Details">
    </form>
</body>
</html>

```

## INTERFACES

# Add Hostel

Hostel Name:

NBA

Address:

CV.RAMAN GLOBAL UNIV

Capacity:

100

Facilities:

GOOD

Add Hostel

# Edit Hostel

Hostel ID:

Hostel Name:

Address:

Capacity:

Facilities:

Edit Hostel

# Delete Hostel

Hostel ID:

Delete Hostel

# Edit Student

Student ID:

2101020306

Name:

Rana sujeet KUMAR

Contact Info:

6203370346

Course:

b.tech

Hostel ID:

1001

Room ID:

302

Edit Student

# Delete Student

Student ID:

2101020306

Delete Student

# View Student Details

Student ID:

# Collect Fee

Student ID:

Amount:

# View Payment History

Student ID:

# Generate Reports

Report Type:

Occupancy Report ▼

Generate Report

Success

Your transaction was successful!

[Go back to Home](#)

## 5 Conclusions

To sum up, this report's Hostel Management System provides a solid means of streamlining hostel management procedures. The system facilitates operations like room allocation, fee payment, and record keeping for both students and administrators through the use of technologies like Java, MySQL, and Apache Tomcat. Efficiency and decision-making skills can be further increased by expanding on current systems and adding suggested improvements, such as mobile application support, interaction with SIS, and analytics tools. The Hostel Management System, with its ongoing upgrades and improvements, continues to be an invaluable tool for managing hostel facilities and makes sure that everyone who uses it has a smooth and productive stay.