Hotel Management System

A Report Submitted in

Partial Fulfilment for

award of Bachelor of Technology/Master of Integrated Technology

In

DBMS (BCSE0452)

COMPUTER SCIENCE & ENGINEERING

By

Muralidhar

(2301331530098)

Under the Supervision of

Ms. SANA ANJUM

Assistant Professor

Department of Computer Science & Engineering

School of Computer Science and Information Technology

NOIDA INSTITUTE OF ENGINEERING AND TECHNOLOGY

GREATER NOIDA

(An Autonomous Institute)

Affiliated to

DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, LUCKNOW

DECLARATION

I hereby declare that the work presented in this report was carried out by me. I have not submitted the matter embodied in this report for the award of any degree or diploma of any other University or Institute.

Student Name:

Muralidhar (Roll No.: 2301331530098)

_____

CERTIFICATE

Certified that **Muralidhar (2301331530098),** has carried out the project work presented in this Project Report in partial fulfillment of the requirements for the award of **Bachelor of Technology/Master of Integrated technology, CSE 2nd Year, 4th Sem** from **Noida Institute of Engineering & Technology** under my supervision.

Signature: _____

Ms. Sana Anjum

Assistant Professor

NIET, Gr. Noida

Signature:_____

Dr. Kumud Saxena

HOD

Department of CSE

NIET, Gr. Noida

ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to **Ms. Sana Anjum** for their invaluable guidance, support, and constant supervision throughout the development of this project. Their expertise and encouragement have been pivotal in shaping the project.

I would also like to extend my thanks and appreciation to our esteemed **HOD (Dr. Kumud Saxena)** for their continuous motivation and support. Their encouragement has been instrumental in completing this project successfully.

Finally, I am deeply thankful to my peers and everyone who directly or indirectly contributed to this project with their knowledge, advice, and assistance.

ABSTRACT

The Hotel Management System is a software application developed using Java in NetBeans IDE with MySQL as the backend database. The primary objective of this project is to automate and streamline the daily operations of a hotel, enhancing both efficiency and customer satisfaction. The system facilitates smooth handling of core hotel functions such as room booking, check-in/check-out processes, customer records management, billing, and payment processing.

This project addresses common challenges faced by hotel staff by replacing traditional manual methods with a reliable and user-friendly interface. It ensures data accuracy, reduces human errors, and provides quick access to essential information. The admin panel allows hotel managers to monitor room availability, manage staff details, generate financial reports, and maintain records efficiently.

The use of Java ensures platform independence, while MySQL provides a robust and secure database structure to store large volumes of data. The system has been designed with scalability and flexibility in mind, making it suitable for small to medium-sized hotels.

In conclusion, the Hotel Management System offers a practical, real-time solution to modernize hotel operations, reduce workload, and improve overall service delivery to guests.

TABLE OF CONTENTS

CHAPTER-1

INTRODUCTION

1.1 Overview

The **Hotel Management System** is a desktop-based application developed using **Java (NetBeans IDE)** and **MySQL** as the backend database. This system is designed to manage and automate all hotel operations efficiently. The main purpose of the project is to simplify the tasks of hotel staff and provide a seamless experience to the guests by digitizing core hotel activities such as room booking, guest check-in/check-out, staff management, and billing.

The application includes modules for different users like the administrator and reception staff. The admin module can manage room types, pricing, employee records, and view system-generated reports. Receptionists can check room availability, register new customers, and handle billing and payments. The system also stores and retrieves customer details for future reference and better service.

By integrating **Java GUI** for the frontend and **MySQL** for data storage, the system ensures a secure and responsive interface. All data is managed in real-time, providing up-to-date information on bookings, room status, and guest history.

Overall, this project aims to increase productivity, reduce manual workload, minimize errors, and enhance customer satisfaction through a digital solution tailored for hotel management.

1.2 Objective and Scope

Objective: The main objectives of the Hotel Management System project are:

1. To automate the routine tasks involved in hotel operations such as booking, check-in/check-out, and billing.
2. To create a user-friendly interface for hotel staff to manage guest information and room availability.
3. To maintain accurate and organized records of customers, staff, and financial transactions.
4. To reduce manual errors and increase efficiency in hotel operations.
5. To provide real-time data access and reporting features for better decision-making.
6. To ensure data security and proper user authentication for system access.

Scope: The scope of this project includes:

- **Room Booking Management**: Customers can be assigned available rooms based on preferences and availability.
- **Customer Management**: Store and retrieve customer details for check-in, check-out, and future references.
- **Billing System**: Automatically calculate bills based on room type, stay duration, and additional services.
- **Employee Management**: Admin can manage hotel staff details, including their roles and shifts.
- **Reports Generation**: Generate various reports like occupancy report, revenue report, and customer records.
- **Data Storage**: MySQL is used to store all data securely and efficiently.
- **Desktop Application**: Built using Java (NetBeans), making it suitable for offline use in small to medium-sized hotels.

CHAPTER-2

LITERATURE REVIEW

2.1 Overview of Existing Systems

In many hotels, the existing system is still **manual or semi-computerized**, which involves a lot of paperwork and time-consuming processes. Room bookings, customer check-ins, billing, and record-keeping are often handled using **registers or basic spreadsheet software** like MS Excel. These methods are not only outdated but also prone to **human errors**, **data loss**, and **inefficiency**.

Hotel staff have to manually check room availability, which can lead to double bookings or incorrect information being provided to customers. Searching for past customer records becomes difficult due to poor data organization. Billing is also done manually, increasing the chances of miscalculations. There is little to no integration between departments, which slows down operations and affects customer satisfaction.

Some hotels may use basic software, but those systems often lack key features such as real-time data updates, proper user access control, or report generation. These limitations create bottlenecks in operations, especially during peak hours.

Overall, the existing system does not support the growing needs of the hospitality industry, which demands **speed, accuracy, and customer-focused services**. Therefore, there's a strong need to shift to an automated and integrated system like the one proposed in this project.

2.2 Limitations of Existing Systems

1. **Manual Errors**: Since most operations are done manually, there are high chances of mistakes in booking, billing, and record-keeping.
2. **Time-Consuming**: Tasks like checking room availability, updating guest records, and generating bills take a lot of time, especially during peak hours.
3. **Lack of Real-Time Data**: Manual or spreadsheet-based systems do not provide real-time updates, leading to issues like double booking or incorrect room status.
4. **Poor Data Management**: Storing and retrieving customer or staff information is difficult due to unorganized records, leading to delays and inconvenience.
5. **Limited Accessibility**: Information is not centralized or digitalized, so only certain staff can access data, and only from specific locations.

6. **No Backup or Recovery**: Paper-based systems or basic spreadsheets usually have no data backup, so there is a high risk of **data loss** due to damage, theft, or system crash.
7. **Security Issues**: Manual records and open files are easily accessible, leading to a lack of data privacy and poor security.
8. **Lack of Reporting and Analytics**: The existing system does not support automatic report generation for occupancy, revenue, or customer trends, which affects management decisions.

CHAPTER-3

REQUIREMENTS

3.1 Functional Requirements

1. **User Authentication**
   - System must allow login for admin and staff users with valid credentials.
   - Password protection and access control for different roles.
2. **Room Management**
   - Add, update, delete room details (room number, type, price, availability).
   - View current room status (occupied/vacant/reserved).
3. **Customer Management**
   - Register new customers with personal and booking details.
   - Update or delete customer information.
   - View booking history of returning customers.
4. **Room Booking**
   - Book available rooms for customers.
   - Assign room based on type and availability.
   - Modify or cancel bookings.
5. **Check-In and Check-Out**
   - Check in a customer and mark the room as occupied.
   - Check out customer, calculate total bill, and mark room as available again.
6. **Billing and Payment**
   - Auto-generate bill based on room type, stay duration, and extra services.
   - Apply taxes and discounts if applicable.
   - Support for multiple payment methods (cash, card, etc.).
7. **Employee Management**
   - Add and manage staff details such as name, role, shift timings, etc.
   - Assign specific roles like receptionist, housekeeping, admin.
8. **Reports and Analytics**
   - Generate reports like daily check-ins, occupancy rate, revenue reports.
   - Export reports for analysis or printing.
9. **Search and Filter Options**
   - Allow searching of customers, bookings, and rooms using filters like name, date, room number, etc.
10. **Database Management**
    - All data (customer, room, booking, employee) to be stored in MySQL database securely.
    - Backup and restore database functionality.

3.2 Non-Functional Requirements.

1. **Performance Requirements**
    - The system should provide quick response times (less than 2 seconds) for actions like booking, check-in, and billing.
    - It should be able to handle multiple users (staff/admin) simultaneously without slowing down.
2. **Reliability**
    - The system should function accurately without failures or crashes.
    - In case of unexpected shutdowns, data integrity must be maintained.
3. **Scalability**
    - The system should be designed in a way that new features (like online booking or mobile app support) can be added in the future.
    - It should support increasing data (more rooms, customers, staff) without performance degradation.
4. **Security**
    - User login credentials must be encrypted.
    - Only authorized users should be able to access specific features based on their roles.
    - Sensitive data (like customer info, payment details) should be protected from unauthorized access.
5. **Usability**
    - The system should have a simple, clean, and user-friendly interface so that staff with basic computer knowledge can use it easily.
    - Proper validation messages and tooltips should be provided for better user experience.
6. **Maintainability**
    - The code and system structure should be modular so that future updates and bug fixes can be done easily.
    - Proper documentation should be provided for reference.
7. **Portability**

- The system should run smoothly on different Windows operating systems.
- It should be easily installable on other desktop systems with Java and MySQL setup.

8. **Backup and Recovery**
    - The system should allow regular data backups.
    - In case of failure or data loss, the system should provide an easy recovery option using the backup.

3.3 Software and Hardware Requirements

Component	Specification
Processor (CPU)	Intel Core i3 or higher
RAM	Minimum 4 GB (8 GB recommended)
Hard Disk	Minimum 250 GB (for data storage)
Monitor	14” or larger, HD resolution
Keyboard & Mouse	Standard input devices
Printer (Optional)	For printing bills and reports
UPS (Optional)	To prevent data loss during power cuts

Software	Version/Details
Operating System	Windows 7/8/10/11
Java Development Kit (JDK)	JDK 8 or higher
NetBeans IDE	Version 8.2 or above
MySQL Database Server	MySQL 5.7 / 8.0

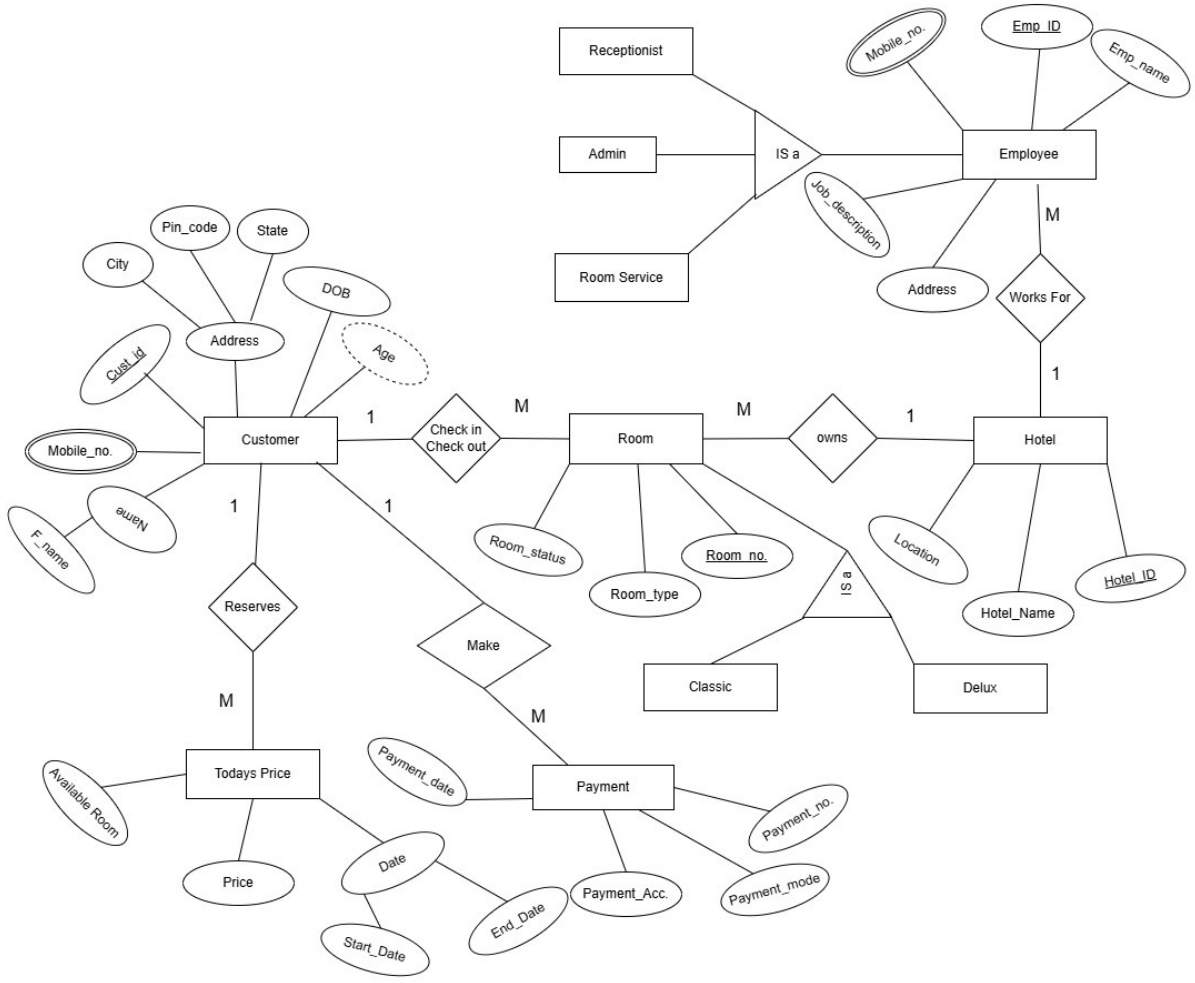
Software	Version/Details
MySQL Workbench	For database design & queries
XAMPP / WAMP	If you want to manage MySQL easily
Microsoft Office / LibreOffice	For documentation and report writing

CHAPTER-4

SYSTEM DESIGN AND IMPLEMENTATION

4.1 System Architecture

FLOWCHART



4.2 GUI Design

The Hotel Management System uses a **Graphical User Interface (GUI)** built with **Java Swing** in **NetBeans IDE**. The interface is designed to be user-friendly, clean, and easy to navigate for hotel staff and administrators. Below is an overview of the major GUI components/screens:

1. Login Screen

- **Fields**: Username, Password
- **Buttons**: Login, Reset
- **Functionality**: Authenticates user and redirects to respective dashboard (Admin or Receptionist)

2. Dashboard (Home Page)

- **Menu Buttons**:
  - Customer Management
  - Room Management
  - Booking
  - Check-In / Check-Out
  - Billing
  - Reports
  - Logout

3. Customer Management Panel

- **Fields**: Name, ID Proof, Contact Number, Address, Email
- **Buttons**: Add Customer, Update, Delete, Search
- **Table View**: Display list of all customers with details

4. Room Management Panel

- **Fields**: Room Number, Room Type, Bed Type, Price, Status (Available/Occupied)
- **Buttons**: Add Room, Edit Room, Delete Room
- **Table View**: Displays all rooms with current status

5. Booking Panel

- **Fields**: Select Customer, Room Type, Room No, Check-in & Check-out Date
- **Buttons**: Book Room, Cancel Booking
- **Table View**: List of current bookings

6. Check-In / Check-Out Panel

- **Fields**: Booking ID or Customer ID
- **Buttons**: Check-In, Check-Out
- **Display**: Room number, Duration of stay, Assigned room status

7. Billing Panel

- **Fields**: Room Charges, Extra Services, Tax, Discounts
- **Buttons**: Generate Bill, Print Bill
- **Display**: Final Amount with breakdown

8. Reports Panel

- **Options**: Generate occupancy report, revenue report, customer history
- **Export Options**: PDF, Excel

Design Notes

- Layout: Using panels, tabbed panes, and layout managers like GridLayout, BorderLayout, etc.
- Color Theme: Light background with clearly labeled buttons and fields
- Navigation: Menu-driven with icons or buttons on the dashboard
- Validation: Input fields have basic validation (e.g., no empty fields, proper format)

4.4 Implementation Details

The Hotel Management System is implemented as a **desktop-based application** using **Java** and the **NetBeans IDE**. The application interacts with a **MySQL database** to manage and store all hotel-related data securely and efficiently. Below is a breakdown of the implementation details:

1. Programming Language

- The system is developed in **Java** using **Swing** for GUI development.
- Java provides object-oriented design and platform independence.

2. Development Environment

- **NetBeans IDE 8.2** is used for writing, compiling, and running the Java code.
- Swing GUI Builder (Matisse) in NetBeans is used to design the interface.

3. Database

- **MySQL** is used for storing all application data such as:
    - Customer details
    - Room records
    - Booking and billing data
    - Employee information
- Tables are normalized to avoid redundancy and maintain data integrity.

4. Database Connectivity

- **JDBC (Java Database Connectivity)** is used to connect Java application to the MySQL database.
- Connection, PreparedStatement, and ResultSet classes are used for querying and updating the database.

5. Application Modules

- **Login Module**: Verifies username and password from the database.
- **Room Management Module**: Allows admin to add/edit/delete room details.
- **Customer Management Module**: Handles customer registration and updates.
- **Booking Module**: Assigns rooms based on availability and records check-in/check-out.
- **Billing Module**: Calculates stay cost, adds services, and generates bill.
- **Reports Module**: Generates reports for occupancy, customer list, and income.

6. Error Handling

- Try-catch blocks are used to handle runtime exceptions such as:
    - Database connection failure
    - Invalid user inputs
    - Null or empty values

7. Security Measures

- Basic login validation with password masking.
- Restricted access based on user roles (e.g., admin vs receptionist).
- SQL queries are written using PreparedStatement to prevent SQL injection.

CHAPTER-5

TESTING

5.1 Testing Methodology

Testing is a crucial phase in software development to ensure that the system functions correctly, meets the requirements, and is free from critical bugs. The Hotel Management System underwent several types of testing to validate its performance, reliability, and user experience.

1. Unit Testing

- Individual modules such as login, booking, billing, and check-in/check-out were tested separately.
- Java methods and database interactions were tested using test data to verify correct output.
- Example: Testing the login method with both valid and invalid credentials.

2. Integration Testing

- After unit testing, modules were integrated and tested together.
- Checked data flow between modules like booking → check-in → billing.
- Ensured that database updates reflect across all related modules correctly.

3. System Testing

- The complete application was tested as a whole.
- Focused on verifying that all requirements (functional and non-functional) are met.
- Included testing of user roles (admin, staff) and overall user workflow.

4. GUI Testing

- Verified that all buttons, fields, and forms work as expected.
- Checked layout consistency and responsiveness across different screen sizes.

- Ensured error messages, tooltips, and validations are properly displayed.

5. Database Testing

- Checked for successful storage and retrieval of data from MySQL database.
- Tested insertion, update, deletion, and selection queries for accuracy and security.
- Validated foreign key constraints and relationships between tables.

6. Performance Testing

- Simulated multiple bookings/check-ins to check system speed and response time.
- Evaluated application behavior under load with simultaneous operations.

7. Security Testing

- Tested login module for invalid access attempts.
- Ensured role-based access control is implemented properly.
- Verified that sensitive data is not exposed to unauthorized users.

Testing Tools Used

- Manual testing for GUI and user interaction.
- Print statements and logs for debugging.
- MySQL Workbench for direct database validation.

5.2 Test Cases and Results

Test Case ID	Test Case Description	Input Data	Expected Output	Actual Result
TC001	Login with valid credentials	Username: admin, Password: admin123	Dashboard loads successfully	Dashboard opened

Test Case ID	Test Case Description	Input Data	Expected Output	Actual Result
TC002	Login with invalid credentials	Username: admin, Password: wrongpass	Error message: "Invalid username or password"	Error message displayed
TC003	Add new room	Room No: 101, Type: AC, Price: 2000	Room added successfully	Room added in database
TC004	Add customer without name	Name: , Contact: 9999999999, ID: A1234	Error message: "Name is required"	Error shown
TC005	Book available room	Customer ID: C101, Room No: 101	Room booked, status updated to occupied	Room booked successfully
TC006	Generate bill after checkout	Room No: 101, Stay: 2 days, Rate: 2000/day	Total: ₹4000 + taxes	Correct bill generated
TC007	Delete customer record	Customer ID: C105	Confirmation prompt → record deleted	Record removed from database
TC008	View occupancy report	Report Type: Occupancy, Date: Last 7 days	Report generated and displayed	Report displayed correctly
TC009	SQL Injection Test (Login)	Username: ' OR '1'='1, Password: xyz	Reject login attempt	Login blocked
TC010	Check room availability filter	Room Type: Non-AC, Status: Available	List of available Non-AC rooms displayed	Filtered result shown

CHAPTER-6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

- The Hotel Management System developed using **Java (NetBeans IDE)** and **MySQL** provides an efficient and user-friendly solution for managing hotel operations. The system automates key functionalities such as room booking, customer management, check-in/check-out processes, billing, and reporting. It reduces manual workload, minimizes errors, and increases the overall productivity of hotel staff.
- Through the implementation of a clean graphical user interface and secure database connectivity, the system ensures smooth navigation and reliable data handling. Features like real-time room status, customer history tracking, and automated bill generation significantly enhance the operational workflow of hotel management.
- This project has been thoroughly tested through unit, integration, and system-level testing, and all functionalities have been verified to work as expected. The system is scalable and maintainable, and it can be extended in the future to include features like online booking, SMS/email notifications, and mobile app integration.
- In conclusion, this project successfully meets its objectives and demonstrates the practical application of software development concepts in the real-world hospitality industry.

.

6.2 Future Work

Although the current version of the Hotel Management System successfully automates various hotel operations, there are several enhancements and features that can be considered for future development:

1. **Online Booking Portal**
    - Integrating an online reservation system that allows customers to book rooms directly through a website or mobile application.
2. **Mobile Application**
    - Developing an Android/iOS app for easier access to booking, billing, and customer management features on mobile devices.
3. **Payment Gateway Integration**
    - Implementing secure online payment methods such as UPI, credit/debit cards, and net banking for customer convenience.
4. **SMS/Email Notifications**

- Sending automated notifications for booking confirmations, reminders, and invoices via SMS or email.

5. **Multi-Hotel Support**
    - Expanding the system to handle operations of multiple branches or hotels from a single platform.
6. **Advanced Reports & Analytics**
    - Adding data analytics tools for forecasting occupancy rates, generating financial insights, and customer trends.
7. **Biometric Login for Staff**
    - Adding fingerprint or face recognition for more secure and convenient staff login.
8. **Cloud Integration**
    - Migrating the system to the cloud to allow remote access, better scalability, and real-time data synchronization.
9. **Inventory Management**
    - Including inventory tracking for hotel supplies like linen, toiletries, and food stock.
10. **Chatbot for Customer Queries**

- Integrating AI-powered chatbots to handle customer inquiries and assist in bookings.

APPENDIX

Source Code

Front Page :

```
package hotelmanagementsystem;


import javax.swing.*;
import java.awt.*;
import java.awt.event.*;


public class HotelManagementSystem extends JFrame implements ActionListener {
HotelManagementSystem(){
//      setSize(1366, 565);
//      setLocation(100, 100);
setBounds(0, 0, 1600, 1200);
setLayout(null);
ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("icons/first.jpg"));
JLabel image = new JLabel(i1);
image.setBounds(0, 0, 1600, 1200);
add(image);


JLabel text = new JLabel("HOTEL MANAGEMENT SYSTEM");
text.setBounds(20, 650, 1000, 90);
text.setForeground(Color.WHITE);
text.setFont(new Font("serif", Font.PLAIN, 50));
image.add(text);


JButton next = new JButton("Next");
```

```
next.setBounds(1200, 700, 150, 50);
next.setBackground(Color.WHITE);
next.setForeground(Color.MAGENTA);
next.addActionListener(this);
next.setFont(new Font("serif", Font.PLAIN, 24));

image.add(next);



setVisible(true);

while(true) {
text.setVisible(false);
try {
Thread.sleep(500);
} catch (Exception e) {
e.printStackTrace();
}
text.setVisible(true);
try {
Thread.sleep(500);
} catch (Exception e) {
e.printStackTrace();
}
}

}

public void actionPerformed(ActionEvent ae) {
```

```
setVisible(false);
new Login();
}



public static void main(String[] args) {
new HotelManagementSystem();
}

}
```

Login Page :

```
package hotelmanagementsystem;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.sql.*;



public class Login extends JFrame implements ActionListener {

JTextField username;
JPasswordField password;
JButton login, cancel;

Login() {
```

```
getContentPane().setBackground(Color.WHITE);

setLayout(null);

JLabel user = new JLabel("username");
user.setBounds(40,20,100,30);
add(user);

username = new JTextField();
username.setBounds(150,20,150,30);
add(username);

JLabel pass = new JLabel("password");
pass.setBounds(40,70,100,30);
add(pass);

password = new JPasswordField();
password.setBounds(150,70,150,30);
add(password);

login = new JButton("Login");
login.setBounds(40,150,120,30);
login.setBackground(Color.BLACK);
login.setForeground(Color.WHITE);
login.addActionListener(this);
add(login);

cancel = new JButton("Cancel");
cancel.setBounds(180,150,120,30);
```

```
cancel.setBackground(Color.BLACK);
cancel.setForeground(Color.WHITE);
cancel.addActionListener(this);
add(cancel);

ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("icons/second.jpg"));
Image i2 = i1.getImage().getScaledInstance(200,200,Image.SCALE_DEFAULT);
ImageIcon i3 = new ImageIcon(i2);
JLabel image = new JLabel(i3);
image.setBounds(350,10,200,200);
add(image);

setBounds(500,200,600,300);
setVisible(true);



}

public void actionPerformed(ActionEvent ae) {
if (ae.getSource() == login) {
String user = username.getText();
String pass = password.getText();

try {
Conn c = new Conn();

String query = "select * from login where username = '"+user+"' and password = '"+pass+"'";
ResultSet rs = c.s.executeQuery(query);
```

```
if (rs.next()) {
setVisible(false);
new Dashboard();
} else {
JOptionPane.showMessageDialog(null, "Invalid username or password");
setVisible(false);
}

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

} else if (ae.getSource() == cancel) {
setVisible(false);
}

}

public static void main(String[] args) {
new Login();
}

}
```

Dashboard :

```
package hotelmanagementsystem;

import javax.swing.*;
```

```
import java.awt.*;
import java.awt.event.*;

public class Dashboard extends JFrame implements ActionListener{
Dashboard() {
setBounds(0,0,1550,1000);

setLayout(null);

ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("icons/third.jpg"));
Image i2 = i1.getImage().getScaledInstance(1550, 1000,Image.SCALE_DEFAULT);
ImageIcon i3 = new ImageIcon(i2);
JLabel image = new JLabel(i3);
image.setBounds(0, 0, 1550, 1000);
add(image);

JLabel text = new JLabel("THE TAJ GROUP WELCOMES YOU");
text.setBounds(400, 80, 1000, 50);
text.setForeground(Color.WHITE);
text.setFont(new Font("Tahoma", Font.BOLD, 46));
image.add(text);

JMenuBar mb = new JMenuBar();
mb.setBounds(0,0,1550,30);
image.add(mb);

JMenu hotel = new JMenu("HOTEL MANAGEMENT");
hotel.setForeground(Color.RED);
mb.add(hotel);
```

```
JMenu admin = new JMenu("ADMIN");
admin.setForeground(Color.BLUE);
admin.addActionListener(this);
mb.add(admin);

JMenuItem reception = new JMenuItem("RECEPTION");
reception.addActionListener(this);
hotel.add(reception);

JMenuItem addemployee = new JMenuItem("ADD EMPLOYEE");
addemployee.addActionListener(this);
admin.add(addemployee);

JMenuItem addrooms = new JMenuItem("ADD ROOMS");
addrooms.addActionListener(this);
admin.add(addrooms);

JMenuItem adddrivers = new JMenuItem("ADD DRIVERS");
admin.add(adddrivers);



setVisible(true);


}

public void actionPerformed(ActionEvent ae) {
```

```
if (ae.getActionCommand().equals("ADD EMPLOYEE")) {
new AddEmployee();
} else if (ae.getActionCommand().equals("ADD ROOMS")) {
new AddRooms();
}
}


public static void main(String[] args) {
new Dashboard();

}


}
```

Add employee :

```
package hotelmanagementsystem;


import java.awt.EventQueue;


import java.awt.*;
import java.awt.event.*;
import java.sql.*;
import javax.swing.*;

public class AddEmployee extends JFrame{ //Third Frame
```

```
JTextField textField,textField_1,textField_2,textField_3,textField_4,textField_5,textField_6;
JComboBox c1;

public AddEmployee(){
getContentPane().setForeground(Color.BLUE);
getContentPane().setBackground(Color.WHITE);
setTitle("ADD EMPLOYEE DETAILS");

setDefaultCloseOperation(JFrame.HIDE_ON_CLOSE);
setSize(778,486);
getContentPane().setLayout(null);

JLabel Passportno = new JLabel("NAME");
Passportno.setFont(new Font("Tahoma", Font.PLAIN, 17));
Passportno.setBounds(60, 30, 150, 27);
add(Passportno);

textField = new JTextField();
textField.setBounds(200, 30, 150, 27);
add(textField);

JButton Next = new JButton("Submit");
Next.setBounds(200, 420, 150, 30);
Next.setBackground(Color.BLACK);
Next.setForeground(Color.WHITE);
add(Next);

JLabel Pnrno = new JLabel("AGE");
```

```
Pnrno.setFont(new Font("Tahoma", Font.PLAIN, 17));
Pnrno.setBounds(60, 80, 150, 27);
add(Pnrno);

textField_1 = new JTextField();
textField_1.setBounds(200, 80, 150, 27);
add(textField_1);

JLabel Gender = new JLabel("GENDER");
Gender.setFont(new Font("Tahoma", Font.PLAIN, 17));
Gender.setBounds(60, 120, 150, 27);
add(Gender);

JRadioButton NewRadioButton = new JRadioButton("MALE");
NewRadioButton.setBackground(Color.WHITE);
NewRadioButton.setBounds(200, 120, 70, 27);
add(NewRadioButton);

JRadioButton Female = new JRadioButton("FEMALE");
Female.setBackground(Color.WHITE);
Female.setBounds(280, 120, 70, 27);
add(Female);


JLabel Address = new JLabel("JOB");
Address.setFont(new Font("Tahoma", Font.PLAIN, 17));
Address.setBounds(60, 170, 150, 27);
add(Address);
```

```
String course[] = {"Front Desk Clerks","Porters","Housekeeping","Kitchen Staff","Room Service","Waiter/Waitress","Manager","Accountant","Chef"};
c1 = new JComboBox(course);
c1.setBackground(Color.WHITE);
c1.setBounds(200,170,150,30);
add(c1);

JLabel Nationality = new JLabel("SALARY");
Nationality.setFont(new Font("Tahoma", Font.PLAIN, 17));
Nationality.setBounds(60, 220, 150, 27);
add(Nationality);

textField_3 = new JTextField();
textField_3.setBounds(200, 220, 150, 27);
add(textField_3);

JLabel Name = new JLabel("PHONE");
Name.setFont(new Font("Tahoma", Font.PLAIN, 17));
Name.setBounds(60, 270, 150, 27);
add(Name);

textField_4 = new JTextField();
textField_4.setBounds(200, 270, 150, 27);
add(textField_4);

JLabel Phno = new JLabel("AADHAR");
Phno.setFont(new Font("Tahoma", Font.PLAIN, 17));
Phno.setBounds(60, 320, 150, 27);
add(Phno);
```

```
textField_5 = new JTextField();
textField_5.setBounds(200, 320, 150, 27);
add(textField_5);



JLabel email = new JLabel("EMAIL");
email.setFont(new Font("Tahoma", Font.PLAIN, 17));
email.setBounds(60, 370, 150, 27);
add(email);

textField_6 = new JTextField();
textField_6.setBounds(200, 370, 150, 27);
add(textField_6);

setVisible(true);

JLabel AddPassengers = new JLabel("ADD EMPLOYEE DETAILS");
AddPassengers.setForeground(Color.BLUE);
AddPassengers.setFont(new Font("Tahoma", Font.PLAIN, 31));
AddPassengers.setBounds(450, 24, 442, 35);
add(AddPassengers);



ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("icons/tenth.jpg"));
Image i3 = i1.getImage().getScaledInstance(500, 500,Image.SCALE_DEFAULT);
ImageIcon i2 = new ImageIcon(i3);
JLabel image = new JLabel(i2);
image.setBounds(410,80,480,410);
```

```
add(image);



Next.addActionListener(new ActionListener(){
public void actionPerformed(ActionEvent ae){
String name = textField.getText();
String age = textField_1.getText();
String salary = textField_3.getText();
String phone = textField_4.getText();
String aadhar = textField_5.getText();
String email = textField_6.getText();

String gender = null;

if(NewRadioButton.isSelected()){
gender = "male";

}else if(Female.isSelected()){
gender = "female";
}



String job = (String)c1.getSelectedItem();

try {
Conn c = new Conn();
String str = "INSERT INTO employee values( '"+name+"', '"+age+"', '"+gender+"','"+job+"',
'"+salary+"', '"+phone+"','"+aadhar+"', '"+email+"')";
```

```
c.s.executeUpdate(str);
JOptionPane.showMessageDialog(null,"Employee Added successfully");
setVisible(false);

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

setSize(900,600);
setVisible(true);
setLocation(530,200);

}

public static void main(String[] args){
new AddEmployee();
}
}
```

Add Room :

```
package hotelmanagementsystem;

import java.awt.*;
import javax.swing.*;
import javax.swing.border.*;
import java.awt.event.*;
import java.sql.*;
import java.util.*;
```

```
public class AddRooms extends JFrame implements ActionListener{

private JPanel contentPane;
private JTextField t1,t2,t3,t4;
private JComboBox comboBox, comboBox_1, comboBox_2, comboBox_3;
JButton b1,b2;
Choice c1;

public static void main(String[] args) {
new AddRooms().setVisible(true);
}


public AddRooms() {
setBounds(450, 200, 1000, 450);
contentPane = new JPanel();
setContentPane(contentPane);
contentPane.setLayout(null);

ImageIcon i1  = new ImageIcon(ClassLoader.getSystemResource("icons/twelve.jpg"));
Image i3 = i1.getImage().getScaledInstance(500, 300,Image.SCALE_DEFAULT);
ImageIcon i2 = new ImageIcon(i3);
JLabel l15 = new JLabel(i2);
l15.setBounds(400,30,500,370);
add(l15);

JLabel l10 = new JLabel("Add Rooms");
l10.setFont(new Font("Tahoma", Font.BOLD, 18));
```

```
l10.setBounds(194, 10, 120, 22);
contentPane.add(l10);



JLabel l1 = new JLabel("Room Number");
l1.setForeground(new Color(25, 25, 112));
l1.setFont(new Font("Tahoma", Font.BOLD, 14));
l1.setBounds(64, 70, 102, 22);
contentPane.add(l1);


t4 = new JTextField();
t4.setBounds(174, 70, 156, 20);
contentPane.add(t4);


JLabel l2 = new JLabel("Availability");
l2.setForeground(new Color(25, 25, 112));
l2.setFont(new Font("Tahoma", Font.BOLD, 14));
l2.setBounds(64, 110, 102, 22);
contentPane.add(l2);

comboBox = new JComboBox(new String[] { "Available", "Occupied" });
comboBox.setBounds(176, 110, 154, 20);
contentPane.add(comboBox);


JLabel l3 = new JLabel("Cleaning Status");
```

```
l3.setForeground(new Color(25, 25, 112));
l3.setFont(new Font("Tahoma", Font.BOLD, 14));
l3.setBounds(64, 150, 102, 22);
contentPane.add(l3);

comboBox_2 = new JComboBox(new String[] { "Cleaned", "Dirty" });
comboBox_2.setBounds(176, 150, 154, 20);
contentPane.add(comboBox_2);

JLabel l4 = new JLabel("Price");
l4.setForeground(new Color(25, 25, 112));
l4.setFont(new Font("Tahoma", Font.BOLD, 14));
l4.setBounds(64, 190, 102, 22);
contentPane.add(l4);

t2 = new JTextField();
t2.setBounds(174, 190, 156, 20);
contentPane.add(t2);

JLabel l5 = new JLabel("Bed Type");
l5.setForeground(new Color(25, 25, 112));
l5.setFont(new Font("Tahoma", Font.BOLD, 14));
l5.setBounds(64, 230, 102, 22);
contentPane.add(l5);


comboBox_3 = new JComboBox(new String[] { "Single Bed", "Double Bed"});
comboBox_3.setBounds(176, 230, 154, 20);
contentPane.add(comboBox_3);
```

```
b1 = new JButton("Add");
b1.addActionListener(this);
b1.setBounds(64, 321, 111, 33);
b1.setBackground(Color.BLACK);
b1.setForeground(Color.WHITE);
contentPane.add(b1);

b2 = new JButton("Back");
b2.addActionListener(this);
b2.setBounds(198, 321, 111, 33);
b2.setBackground(Color.BLACK);
b2.setForeground(Color.WHITE);
contentPane.add(b2);


contentPane.setBackground(Color.WHITE);

}

public void actionPerformed(ActionEvent ae){
try{

if(ae.getSource() == b1){
try{
```

```
Conn c = new Conn();
String roomnumber = t4.getText();
String availability = (String)comboBox.getSelectedItem();
String cleaning_status = (String)comboBox_2.getSelectedItem();
String price  = t2.getText();
String bed_type = (String)comboBox_3.getSelectedItem();
String str = "INSERT INTO room values( '"+roomnumber+"', '"+availability+"', '"+cleaning_status+"','"+price+"', '"+bed_type+"')";



c.s.executeUpdate(str);
JOptionPane.showMessageDialog(null, "Room Successfully Added");
this.setVisible(false);


}catch(Exception ee){
System.out.println(ee);
}
}
else if(ae.getSource() == b2){
this.setVisible(false);
}
}catch(Exception eee){


}
}
}
```

Add Drivers :

```
package hotelmanagementsystem;
```

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.sql.*;

public class AddDrivers extends JFrame implements ActionListener {

private JPanel contentPane;
private JTextField t1, t2, t3, t4, t5;
private JComboBox<String> comboBox, comboBox_1;
JButton b1, b2;

public static void main(String[] args) {
new AddDrivers().setVisible(true);
}

public AddDrivers() {
setBounds(450, 200, 1000, 500);
contentPane = new JPanel();
contentPane.setLayout(null);
setContentPane(contentPane);

ImageIcon i1 = new ImageIcon(ClassLoader.getSystemResource("icons/eleven.jpg"));
Image i3 = i1.getImage().getScaledInstance(500, 300, Image.SCALE_DEFAULT);
ImageIcon i2 = new ImageIcon(i3);
JLabel l15 = new JLabel(i2);
l15.setBounds(400, 30, 500, 370);
contentPane.add(l15);
```

```
JLabel l10 = new JLabel("Add Drivers");
l10.setFont(new Font("Tahoma", Font.BOLD, 18));
l10.setBounds(194, 10, 120, 22);
contentPane.add(l10);

JLabel l1 = new JLabel("Name");
l1.setFont(new Font("Tahoma", Font.BOLD, 14));
l1.setBounds(64, 70, 102, 22);
contentPane.add(l1);

t1 = new JTextField();
t1.setBounds(174, 70, 156, 20);
contentPane.add(t1);

JLabel l2 = new JLabel("Age");
l2.setFont(new Font("Tahoma", Font.BOLD, 14));
l2.setBounds(64, 110, 102, 22);
contentPane.add(l2);

t2 = new JTextField();
t2.setBounds(174, 110, 156, 20);
contentPane.add(t2);

JLabel l3 = new JLabel("Gender");
l3.setFont(new Font("Tahoma", Font.BOLD, 14));
l3.setBounds(64, 150, 102, 22);
contentPane.add(l3);
```

```
comboBox = new JComboBox<>(new String[]{"Male", "Female"});
comboBox.setBounds(176, 150, 154, 20);
contentPane.add(comboBox);

JLabel l4 = new JLabel("Car Company");
l4.setFont(new Font("Tahoma", Font.BOLD, 14));
l4.setBounds(64, 190, 102, 22);
contentPane.add(l4);

t3 = new JTextField();
t3.setBounds(174, 190, 156, 20);
contentPane.add(t3);

JLabel l5 = new JLabel("Car Brand");
l5.setFont(new Font("Tahoma", Font.BOLD, 14));
l5.setBounds(64, 230, 102, 22);
contentPane.add(l5);

t4 = new JTextField();
t4.setBounds(174, 230, 156, 20);
contentPane.add(t4);

JLabel l6 = new JLabel("Available");
l6.setFont(new Font("Tahoma", Font.BOLD, 14));
l6.setBounds(64, 270, 102, 22);
contentPane.add(l6);

comboBox_1 = new JComboBox<>(new String[]{"Yes", "No"});
comboBox_1.setBounds(176, 270, 154, 20);
```

```
contentPane.add(comboBox_1);

JLabel l7 = new JLabel("Location");
l7.setFont(new Font("Tahoma", Font.BOLD, 14));
l7.setBounds(64, 310, 102, 22);
contentPane.add(l7);

t5 = new JTextField();
t5.setBounds(174, 310, 156, 20);
contentPane.add(t5);



b1 = new JButton("Add");
b1.setBounds(64, 380, 111, 33);
b1.setBackground(Color.BLACK);
b1.setForeground(Color.WHITE);
b1.addActionListener(this);
contentPane.add(b1);

b2 = new JButton("Back");
b2.setBounds(198, 380, 111, 33);
b2.setBackground(Color.BLACK);
b2.setForeground(Color.WHITE);
b2.addActionListener(this);
contentPane.add(b2);

contentPane.setBackground(Color.WHITE);
}
```

```
public void actionPerformed(ActionEvent ae) {
if (ae.getSource() == b1) {
String name = t1.getText();
String age = t2.getText();
String gender = (String) comboBox.getSelectedItem();
String company = t3.getText();
String brand = t4.getText();
String available = (String) comboBox_1.getSelectedItem();
String location = t5.getText();

try {
Conn c = new Conn();

String str = "INSERT INTO driver VALUES('" + name + "','" + age + "','" + gender + "','" +
company + "','" + brand + "','" + available + "','" + location + "')";
c.s.executeUpdate(str);
JOptionPane.showMessageDialog(null, "New Driver Added Successfully");
setVisible(false);
} catch (Exception e) {
e.printStackTrace();
}
} else {
setVisible(false);
}
}
}
```

New Customer :

```
package hotelmanagementsystem;
```

```
import java.awt.BorderLayout;
import java.awt.*;
import java.awt.EventQueue;

import javax.swing.border.EmptyBorder;

import java.awt.Font;
import java.awt.Image;
import java.sql.*;
import javax.swing.*;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class NewCustomer extends JFrame {
Connection conn = null;
PreparedStatement pst = null;
private JPanel contentPane;
private JTextField t1,t2,t3,t4,t5,t6;
JComboBox comboBox;
JRadioButton r1,r2;
Choice c1;
/**
* Launch the application.
*/
public static void main(String[] args) {
EventQueue.invokeLater(new Runnable() {
public void run() {
try {
```

```
NewCustomer frame = new NewCustomer();
frame.setVisible(true);
} catch (Exception e) {
e.printStackTrace();
}
}
});
}

public NewCustomer() throws SQLException {

setBounds(530, 200, 850, 550);
contentPane = new JPanel();
setContentPane(contentPane);

contentPane.setLayout(null);

ImageIcon i1  = new ImageIcon(ClassLoader.getSystemResource("icons/fifth.png"));
Image i3 = i1.getImage().getScaledInstance(300, 400,Image.SCALE_DEFAULT);
ImageIcon i2 = new ImageIcon(i3);
JLabel l1 = new JLabel(i2);
l1.setBounds(480,10,300,500);
add(l1);

JLabel lblName = new JLabel("NEW CUSTOMER FORM");
lblName.setFont(new Font("Yu Mincho", Font.PLAIN, 20));
lblName.setBounds(118, 11, 260, 53);
contentPane.add(lblName);
```

```
JLabel lblId = new JLabel("ID :");
lblId.setBounds(35, 76, 200, 14);
contentPane.add(lblId);

comboBox = new JComboBox(new String[] {"Passport", "Aadhar Card", "Voter Id", "Driving license"});
comboBox.setBounds(271, 73, 150, 20);
contentPane.add(comboBox);

JLabel l2 = new JLabel("Number :");
l2.setBounds(35, 111, 200, 14);
contentPane.add(l2);

t1 = new JTextField();
t1.setBounds(271, 111, 150, 20);

contentPane.add(t1);
t1.setColumns(10);

JLabel lblName_1 = new JLabel("Name :");
lblName_1.setBounds(35, 151, 200, 14);
contentPane.add(lblName_1);

t2 = new JTextField();
t2.setBounds(271, 151, 150, 20);
contentPane.add(t2);
t2.setColumns(10);
```

```
JLabel lblGender = new JLabel("Gender :");
lblGender.setBounds(35, 191, 200, 14);
contentPane.add(lblGender);

r1 = new JRadioButton("Male");
r1.setFont(new Font("Raleway", Font.BOLD, 14));
r1.setBackground(Color.WHITE);
r1.setBounds(271, 191, 80, 12);
add(r1);

r2 = new JRadioButton("Female");
r2.setFont(new Font("Raleway", Font.BOLD, 14));
r2.setBackground(Color.WHITE);
r2.setBounds(350, 191, 100, 12);
add(r2);


JLabel lblCountry = new JLabel("Country :");
lblCountry.setBounds(35, 231, 200, 14);
contentPane.add(lblCountry);

JLabel lblReserveRoomNumber = new JLabel("Allocated Room Number :");
lblReserveRoomNumber.setBounds(35, 274, 200, 14);
contentPane.add(lblReserveRoomNumber);

c1 = new Choice();
try{
Conn c = new Conn();
ResultSet rs = c.s.executeQuery("select * from room where availability = 'available'");
```

```
while(rs.next()){
c1.add(rs.getString("roomnumber"));
}
}catch(Exception e){ }
c1.setBounds(271, 274, 150, 20);
contentPane.add(c1);

JLabel lblCheckInStatus = new JLabel("Checked-In :");
lblCheckInStatus.setBounds(35, 316, 200, 14);
contentPane.add(lblCheckInStatus);

JLabel lblDeposite = new JLabel("Deposit :");
lblDeposite.setBounds(35, 359, 200, 14);
contentPane.add(lblDeposite);




t3 = new JTextField();
t3.setBounds(271, 231, 150, 20);
contentPane.add(t3);
t3.setColumns(10);



t5 = new JTextField();
t5.setBounds(271, 316, 150, 20);
contentPane.add(t5);
t5.setColumns(10);
```

```
t6 = new JTextField();
t6.setBounds(271, 359, 150, 20);
contentPane.add(t6);
t6.setColumns(10);

JButton btnNewButton = new JButton("Add");
btnNewButton.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
Conn c = new Conn();
String radio = null;

if(r1.isSelected()){
radio = "Male";
}
else if(r2.isSelected()){
radio = "Female";
}

String s6 = c1.getSelectedItem();

try{

String s1 = (String)comboBox.getSelectedItem();
String s2 =  t1.getText();
String s3 =  t2.getText();
String s4 =  radio;
String s5 =  t3.getText();
String s7 =  t5.getText();
```

```
String s8 =  t6.getText();

String q1 = "insert into customer values('"+s1+"','"+s2+"','"+s3+"','"+s4+"','"+s5+"','"+s6+"','"+s7+"','"+s8+"')";
String q2 = "update room set availability = 'Occupied' where roomnumber = "+s6;
c.s.executeUpdate(q1);
c.s.executeUpdate(q2);

JOptionPane.showMessageDialog(null, "New Customer Added Successfully");
new Reception().setVisible(true);
setVisible(false);
}catch(SQLException e1){
System.out.println(e1.getMessage());
}
catch(NumberFormatException s){
JOptionPane.showMessageDialog(null, "Please enter a valid Number");
}
}
});
btnNewButton.setBounds(100, 430, 120, 30);
btnNewButton.setBackground(Color.BLACK);
btnNewButton.setForeground(Color.WHITE);
contentPane.add(btnNewButton);

JButton btnExit = new JButton("Back");
btnExit.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
new Reception().setVisible(true);
```

```
setVisible(false);
}
});
btnExit.setBounds(260, 430, 120, 30);
btnExit.setBackground(Color.BLACK);
btnExit.setForeground(Color.WHITE);
contentPane.add(btnExit);

getContentPane().setBackground(Color.WHITE);
}
}
```

Department :

```
package hotelmanagementsystem;

import java.awt.BorderLayout;
import java.awt.*;

import javax.swing.JFrame;
import javax.swing.JPanel;
import javax.swing.border.EmptyBorder;

import net.proteanit.sql.DbUtils;

import javax.swing.JTable;
import java.sql.*;
import javax.swing.*;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
```

```
public class Department extends JFrame {
Connection conn = null;
private JPanel contentPane;
private JTable table;
private JLabel lblNewLabel;
private JLabel lblNewLabel_1;

/**
* Launch the application.
*/
public static void main(String[] args) {
EventQueue.invokeLater(new Runnable() {
public void run() {
try {
Department frame = new Department();
frame.setVisible(true);
} catch (Exception e) {

e.printStackTrace();
}
}
});
}
public void close()
{
this.dispose();
}

/**
```

```
* Create the frame.
* @throws SQLException
*/
public Department() throws SQLException {
//conn = Javaconnect.getDBConnection();
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setBounds(600, 200, 700, 500);
contentPane = new JPanel();
contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
setContentPane(contentPane);
contentPane.setLayout(null);

table = new JTable();
table.setBounds(0, 40, 700, 350);
contentPane.add(table);

JButton btnNewButton = new JButton("Load Data");
btnNewButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {
try{
Conn c = new Conn();

String displayCustomersql = "select * from Department";
ResultSet rs = c.s.executeQuery(displayCustomersql);
table.setModel(DbUtils.resultSetToTableModel(rs));



}
```

```
catch (Exception e1){
e1.printStackTrace();
}

}
});
btnNewButton.setBounds(170, 410, 120, 30);
btnNewButton.setBackground(Color.BLACK);
btnNewButton.setForeground(Color.WHITE);
contentPane.add(btnNewButton);

JButton btnNewButton_1 = new JButton("Back");
btnNewButton_1.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
new Reception().setVisible(true);
setVisible(false);
}
});
btnNewButton_1.setBounds(400, 410, 120, 30);

btnNewButton_1.setBackground(Color.BLACK);
btnNewButton_1.setForeground(Color.WHITE);
contentPane.add(btnNewButton_1);

lblNewLabel = new JLabel("Department");
lblNewLabel.setBounds(145, 11, 105, 14);
contentPane.add(lblNewLabel);

lblNewLabel_1 = new JLabel("Budget");
```

```
lblNewLabel_1.setBounds(431, 11, 75, 14);
contentPane.add(lblNewLabel_1);

getContentPane().setBackground(Color.WHITE);
}

}
```

Reception :

```
package hotelmanagementsystem;

import javax.swing.*;

import java.sql.*;
import java.awt.event.*;
import java.awt.*;

public class Reception extends JFrame {



private JPanel contentPane;

public static void main(String[] args) {
new Reception();
}

public Reception(){
```

```
setBounds(530, 200, 850, 570);
contentPane = new JPanel();
setContentPane(contentPane);
contentPane.setLayout(null);

ImageIcon i1  = new ImageIcon(ClassLoader.getSystemResource("icons/fourth.jpg"));
Image i3 = i1.getImage().getScaledInstance(500, 500,Image.SCALE_DEFAULT);
ImageIcon i2 = new ImageIcon(i3);
JLabel l1 = new JLabel(i2);
l1.setBounds(250,30,500,470);
add(l1);

JButton btnNewCustomerForm = new JButton("New Customer Form");
btnNewCustomerForm.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
try{
NewCustomer custom = new NewCustomer();
custom.setVisible(true);
setVisible(false);
}catch(Exception e1){
e1.printStackTrace();
}
}
});
btnNewCustomerForm.setBounds(10, 30, 200, 30);
btnNewCustomerForm.setBackground(Color.BLACK);
btnNewCustomerForm.setForeground(Color.WHITE);
contentPane.add(btnNewCustomerForm);
```

```
JButton btnNewButton = new JButton("Room");
btnNewButton.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
try{
Room room = new Room();
room.setVisible(true);
setVisible(false);
}
catch(Exception e){
e.printStackTrace();
}

}
});
btnNewButton.setBounds(10, 70, 200, 30);
btnNewButton.setBackground(Color.BLACK);
btnNewButton.setForeground(Color.WHITE);

contentPane.add(btnNewButton);

JButton btnNewButton_1 = new JButton("Department");

btnNewButton_1.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
try{
Department dept = new Department();
dept.setVisible(true);
setVisible(false);
```

```
}
catch (Exception e1){
e1.printStackTrace();
}

}
});
btnNewButton_1.setBounds(10, 110, 200, 30);
btnNewButton_1.setBackground(Color.BLACK);
btnNewButton_1.setForeground(Color.WHITE);

contentPane.add(btnNewButton_1);

JButton btnNewButton_2 = new JButton("All Employee Info");
btnNewButton_2.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
try{

EmployeeInfo em = new EmployeeInfo();
em.setVisible(true);
setVisible(false);

}
catch (Exception e1){
e1.printStackTrace();
}

}
});
```

```
btnNewButton_2.setBounds(10, 150, 200, 30);
btnNewButton_2.setBackground(Color.BLACK);
btnNewButton_2.setForeground(Color.WHITE);

contentPane.add(btnNewButton_2);

JButton btnNewButton_3 = new JButton("Customer Info");
btnNewButton_3.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
try{
//error ----------------------
CustomerInfo customer = new CustomerInfo();
customer.setVisible(true);
setVisible(false);
}
catch (Exception e1){
e1.printStackTrace();
}
}
});
btnNewButton_3.setBounds(10, 190, 200, 30);
btnNewButton_3.setBackground(Color.BLACK);

btnNewButton_3.setForeground(Color.WHITE);

contentPane.add(btnNewButton_3);

JButton btnManagerInfo = new JButton("Manager Info");
btnManagerInfo.addActionListener(new ActionListener() {
```

```
public void actionPerformed(ActionEvent e) {
try{
ManagerInfo mana = new ManagerInfo();
mana.setVisible(true);
setVisible(false);
}
catch (Exception e1){
e1.printStackTrace();
}
}
});
btnManagerInfo.setBounds(10, 230, 200, 30);
btnManagerInfo.setBackground(Color.BLACK);
btnManagerInfo.setForeground(Color.WHITE);



contentPane.add(btnManagerInfo);

JButton btnNewButton_4 = new JButton("Check Out");
btnNewButton_4.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
CheckOut check;
try {

check = new CheckOut();
check.setVisible(true);
setVisible(false);
} catch (SQLException e1) {
// TODO Auto-generated catch block
```

```
e1.printStackTrace();
}
}
});
btnNewButton_4.setBounds(10, 270, 200, 30);
btnNewButton_4.setBackground(Color.BLACK);
btnNewButton_4.setForeground(Color.WHITE);

contentPane.add(btnNewButton_4);

JButton btnNewButton_5 = new JButton("Update Check Status");
btnNewButton_5.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
try{
UpdateCheck update = new UpdateCheck();
update.setVisible(true);
setVisible(false);
}
catch(Exception e1){
e1.printStackTrace();
}
}
});
btnNewButton_5.setBounds(10, 310, 200, 30);

btnNewButton_5.setBackground(Color.BLACK);
btnNewButton_5.setForeground(Color.WHITE);

contentPane.add(btnNewButton_5);
```

```
JButton btnNewButton_6 = new JButton("Update Room Status");
btnNewButton_6.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
try{
UpdateRoom room = new UpdateRoom();
room.setVisible(true);
setVisible(false);
}catch(Exception s)
{
s.printStackTrace();
}
}
});
btnNewButton_6.setBounds(10, 350, 200, 30);
btnNewButton_6.setBackground(Color.BLACK);
btnNewButton_6.setForeground(Color.WHITE);

contentPane.add(btnNewButton_6);

JButton btnPickUpSerice = new JButton("Pick up Service");
btnPickUpSerice.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent arg0) {
try{
PickUp pick = new PickUp();

pick.setVisible(true);
setVisible(false);
}
```

```
catch(Exception e){
e.printStackTrace();
}
}
});
btnPickUpSerice.setBounds(10, 390, 200, 30);
btnPickUpSerice.setBackground(Color.BLACK);
btnPickUpSerice.setForeground(Color.WHITE);

contentPane.add(btnPickUpSerice);

JButton btnSearchRoom = new JButton("Search Room");
btnSearchRoom.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
try {
SearchRoom search = new SearchRoom();
search.setVisible(true);
setVisible(false);
}
catch (Exception ss){
ss.printStackTrace();
}
}
});
btnSearchRoom.setBounds(10, 430, 200, 30);
btnSearchRoom.setBackground(Color.BLACK);

btnSearchRoom.setForeground(Color.WHITE);
```

```
contentPane.add(btnSearchRoom);

JButton btnNewButton_7 = new JButton("Log Out");
btnNewButton_7.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent ae) {
try {
new Login().setVisible(true);
setVisible(false);

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

}
});
btnNewButton_7.setBounds(10, 470, 200, 30);
btnNewButton_7.setBackground(Color.BLACK);
btnNewButton_7.setForeground(Color.WHITE);

contentPane.add(btnNewButton_7);
getContentPane().setBackground(Color.WHITE);

setVisible(true);
}
}
```

Databases :

```
create database hotelmanagementsystem;
```

```
show databases;

use hotelmanagementsystem;

create table login(username varchar(25), password varchar(25));

insert into login values('admin', '12345');
insert into login values('pradeep', '123456');
insert into login values('murali', '12345');

select * from login;

create table employee(name varchar(25), age  varchar(15), gender varchar(15),
 job varchar(30), salary varchar(15), phone varchar(15), email varchar(40),
 aadhar varchar(20));

 describe employee;

select * from employee;

create table room(roomnumber varchar(10),
availability varchar(20), price varchar(20),
cleaning_status varchar(20), bed_type varchar(20));

 select * from room;


 create table driver( name varchar(20), age varchar(10),
```

```
gender varchar(15), company varchar(20), brand varchar(20),
available varchar(20), location varchar(40));

select * from driver;

create table customer( document varchar(20), number varchar(30),
name varchar(30), gender varchar(15), country varchar(20),
room varchar(10), checked_in varchar(80), deposite varchar(20));

select * from customer;

create table department(department varchar(30), budget varchar(30));

insert into department values('Front Office','500000');
insert into department values('Housekeeping', '40000');
insert into department values('Food and Beverage', '23000');
insert into department values('Kitchen or Food Production', '540000');
insert into department values('Security', '32000');
select * from department;

create table customer( document varchar(20), number varchar(30),
name varchar(30), gender varchar(15), country varchar(20),
room varchar(10), checked_in varchar(80), deposite varchar(20));

select * from customer;
```

GitHub Repository Link

https://github.com/Muralidhar8/Hotel-Management-System

References

1. **Oracle Java Documentation**
   https://docs.oracle.com/javase/8/docs/
2. **MySQL 8.0 Reference Manual**
   https://dev.mysql.com/doc/
3. **NetBeans IDE Documentation**
   https://netbeans.apache.org/kb/index.html
4. **Java Swing Tutorial – GeeksforGeeks**
   https://www.geeksforgeeks.org/java-swing/
5. **JDBC Tutorial – TutorialsPoint**
   https://www.tutorialspoint.com/jdbc/index.htm
6. **MySQL Workbench Documentation**
   https://dev.mysql.com/doc/workbench/en/