

HOTEL RESERVATION SYSTEM



A PROJECT REPORT

Submitted by

S.DIKSHA (2303811710422032)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

NOVEMBER- 2024

**K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY
(AUTONOMOUS)**

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on “**HOTEL RESERVATION SYSTEM**” is the bonafide work of **S.DIKSHA (2303811710422032)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING
Dr.A.DELPHIN CAROLINA RANI, M.E., Ph.D.,
HEAD OF THE DEPARTMENT
PROFESSOR

SIGNATURE

Dr.A.Delphin Carolina Rani, M.E., Ph.D.,

HEAD OF THE DEPARTMENT

PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology
(Autonomous)

Samayapuram–621112.

CGB1201-JAVA PROGRAMMING
Mr. M. SARAVANAN, M.E.,
SUPERVISOR
ASSISTANT PROFESSOR

SIGNATURE

Mr. M. Saravanan, M.E.,

SUPERVISOR

ASSISTANT PROFESSOR

Department of CSE

K.Ramakrishnan College of Technology
(Autonomous)

Samayapuram–621112.

Submitted for the viva-voce examination held on 02.12.2024

CGB1201-JAVA PROGRAMMING
Mr. M. ARMANAN A, M.E.,
INTERNAL EXAMINER
ASSISTANT PROFESSOR

INTERNAL EXAMINER

CGB1201-JAVA PROGRAMMING
Dr. R. SETHAMILSELVI, M.E., Ph.D.,
EXTERNAL EXAMINER
PROFESSOR
8138-SCE, TRICHY.

EXTERNAL EXAMINER

DECLARATION

I declare that the project report on “**HOTEL RESERVATION SYSTEM**” is the result of original work done by us and best of our knowledge, similar work has not been submitted to “**ANNA UNIVERSITY CHENNAI**” for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201 - JAVA PROGRAMMING**.

Signature

S. Diksha

DIKSHA.S

Place: Samayapuram

Date: 02.12.2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution “**K.Ramakrishnan College of Technology (Autonomous)**”, for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN, B.E.**, for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.**, for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.**, Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.**, Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mr. M. SARAVANAN, M.E.**, Department of **COMPUTER SCIENCE AND ENGINEERING**, for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of

our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

- 1. Engineering knowledge:** Apply the knowledge of mathematics, science,

engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a

member or leader in diverse teams, and in multidisciplinary settings.

- 10. Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
- 11. Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
- 12. Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

The Hotel Reservation System is a Java-based desktop application that facilitates room reservations in a hotel. Implemented using the AWT (Abstract Window Toolkit), this graphical user interface (GUI) application allows users to reserve rooms based on availability and their requirements. It features a streamlined interface with fields for entering a customer's name, selecting a room type, specifying the number of days for the stay, and buttons for submitting, clearing, or viewing room availability. The application dynamically manages room inventory across three categories: Single, Double, and Suite rooms, ensuring accurate tracking of available rooms. Upon reservation, the system calculates the total cost based on the selected room type and number of days, updates the room availability, and provides a detailed confirmation. If rooms of the chosen type are unavailable, it promptly notifies the user. The "Show Number of Rooms" button provides a real-time update on the current availability of all room types. Error handling is incorporated to ensure valid user inputs, such as checking for empty fields or invalid data types for the number of days. The application's simplicity and functionality make it an efficient tool for basic hotel management tasks, demonstrating core principles of event-driven programming, GUI design, and dynamic data handling in Java.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
The Hotel Reservation System is a comprehensive software solution designed to streamline the process of booking accommodations for both customers and hotel administrators. It facilitates real-time room availability checks, reservation management, and payment processing while ensuring a user-friendly interface for seamless interactions. For administrators, it offers tools for managing room inventory, customer data, pricing, and special offers, enhancing operational efficiency.	PO1 -3 PO2 -3 PO3 -3 PO4 -3 PO5 -3 PO6 -3 PO7 -3 PO8 -3 PO9 -3 PO10 -3 PO11-3 PO12 -3	PSO1 -3 PSO2 -3 PSO3 -3

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	
1	INTRODUCTION	
	1.1 Objective	
	1.2 Overview	
	1.3 Java Programming concepts	
2	PROJECT METHODOLOGY	
	2.1 Proposed Work	
	2.2 Block Diagram	
3	MODULE DESCRIPTION	
	3.1 User Input Module	
	3.2 Room Management Module	
	3.3 Reservation Processing Module	
	3.4 GUI Components Module	
	3.5 Event Handling Module	
4	CONCLUSION & FUTURE SCOPE	
	4.1 Conclusion	
	4.2 Future Scope	
	REFERENCES	
	APPENDIX A (SOURCE CODE)	
	APPENDIX B (SCREENSHOTS)	

CHAPTER 1

INTRODUCTION

1.1 Objective

The objective of the Hotel Reservation System program is to provide a user-friendly and efficient desktop application for managing hotel room reservations. It aims to streamline the reservation process by allowing users to select room types, specify the duration of stay, and calculate the total cost of the booking dynamically. The system ensures real-time management of room availability, reducing the chances of overbooking or manual errors. By leveraging a graphical user interface (GUI) built with Java's AWT, the program enhances usability, enabling users to perform tasks such as viewing room availability, clearing inputs, and receiving detailed reservation confirmations with ease. Additionally, the program emphasizes input validation and error handling, ensuring data integrity and a seamless user experience. The overarching goal is to demonstrate the principles of event-driven programming while providing a functional and practical solution for basic hotel management needs.

1.2 Overview

The Hotel Reservation System program is a comprehensive demonstration of core Java programming concepts, focusing on graphical user interface (GUI) development and event-driven programming. Using Java's Abstract Window Toolkit (AWT), the program creates an intuitive interface that allows users to book hotel rooms by selecting the room type, specifying the number of days, and providing their details. It dynamically updates room availability and calculates the total cost of reservations, showcasing effective data manipulation techniques. Event handling is central to the program, with actions like button clicks triggering validation, reservation processing, and output generation. Error handling ensures a robust user experience, preventing invalid inputs and providing clear feedback. The program serves as a practical example of building real-world applications in Java.

1.3 Java Programming Concepts

1.Object-Oriented Programming (OOP)

- 1.Encapsulation: All variables (e.g., singleRooms, doubleRooms, suiteRooms) and methods are encapsulated within the HotelReservationSystem class.
- 2.Class and Objects: The program uses a single class (HotelReservationSystem) with a constructor to define and initialize the application's behavior.
- 3.Inheritance: The program inherits from Frame (from java.awt), allowing the use of GUI-related properties and methods.
- 4.Polymorphism: The actionPerformed() method demonstrates polymorphism by handling different button events through the same interface (ActionListener).

2. Graphical User Interface (GUI) Programming

- 1.AWT Components: Uses GUI elements like Frame, Label, TextField, Choice, Button, and TextArea.
- 2.Layout Management: Implements FlowLayout to organize components dynamically within the frame.
- 3.Event Handling: Listeners such as ActionListener and WindowAdapter are used to handle user interactions (e.g., button clicks and window closing)

3. Event-Driven Programming

Events are central to the application's behavior.

The actionPerformed() method handles specific actions triggered by the user (e.g., submitting, clearing fields, or showing room availability).

The WindowAdapter class overrides the windowClosing() method to ensure proper window management.

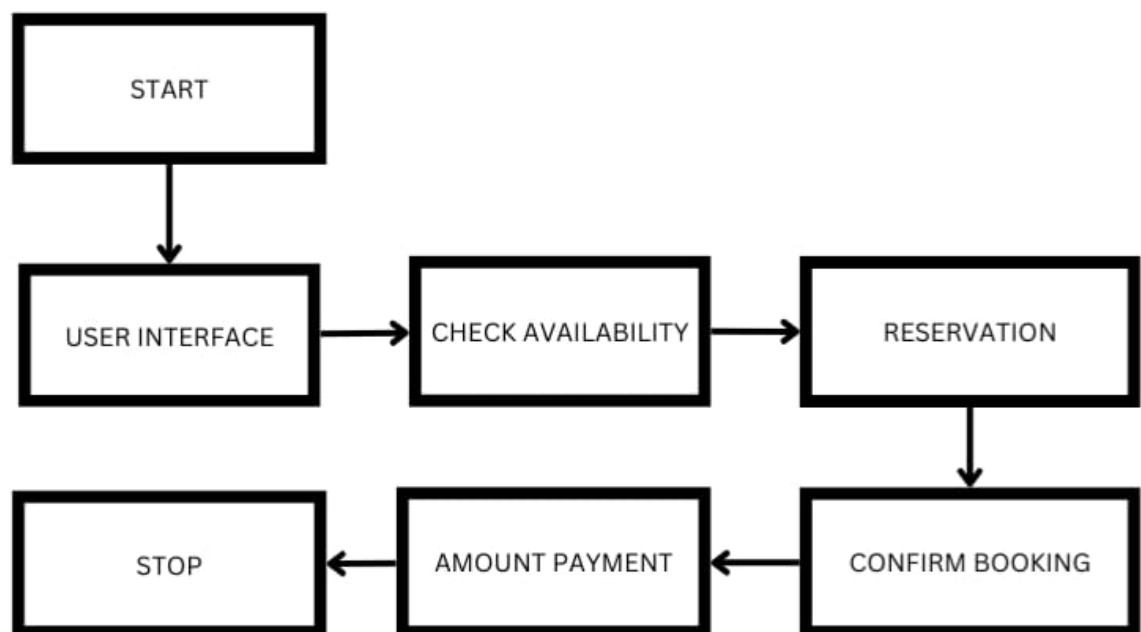
CHAPTER 2

PROJECT METHODOLOGY

2.1 Proposed Work

The proposed work for the Hotel Reservation System program involves creating a user-friendly desktop application that simplifies hotel room booking by dynamically managing room availability, calculating costs, and providing real-time feedback to users. The system is designed to handle user input, validate data, and process reservations efficiently. It integrates GUI components, event-driven logic, and dynamic data handling to achieve a seamless booking experience. The block diagram below outlines the key components and workflows of the program.

2.2 Block Diagram



CHAPTER 3

MODULE DESCRIPTION

3.1 Module 1 : User Input Module

Handles all user input across the application, including validation and preprocessing:

- Collects input from the user via the GUI.
- Validates input data (e.g., dates, room IDs, event details).
- Sends the processed data to relevant modules.

3.2 Module 2 : Room Management Module

Manages information and operations related to rooms:

- Stores room details (types, availability, pricing).
- Provides functionalities to add, update, and remove room records.
- Handles room availability checks and updates

3.3 Module 3 : Reservation Processing Module

Handles reservations and bookings:

- Processes room bookings and event reservations.
- Manages cancellation or modification of reservations.
- Communicates with the Room Management Module to update availability.
- Generates and tracks booking confirmations and invoices.

3.4 Module 4 : GUI Components Module

Defines and manages the graphical user interface:

- Creates windows, forms, buttons, and input fields.

- Displays information such as room details, booking statuses, and errors.

3.5Module 5 : Event Handling Module

Handles user interactions and application events:

- Listens for actions like button clicks, form submissions, or menu selections.
- Connects user actions to backend logic in other modules (e.g., triggering a booking).
- Provides error handling for invalid or incomplete actions.

CHAPTER 4

CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

This program follows a modular design approach, breaking down the system into five core modules. The User Input Module captures and validates user data, ensuring correct input for further processing. The Room Management Module manages room availability, details, and updates. The Reservation Processing Module handles the creation, modification, and cancellation of reservations, updating the room availability accordingly. The GUI Components Module focuses on providing a user-friendly interface, offering intuitive navigation and input fields. The Event Handling Module listens for user interactions and triggers the appropriate actions within the system. By dividing the program into these distinct modules, the design promotes maintainability, scalability, and efficient management of room and reservation data. This modularity also makes the program adaptable to future enhancements or changes in functionality.

4.2 FUTURE SCOPE

The future scope of this modular room and reservation management system can include several enhancements to improve functionality, scalability, and user experience.

- 1.Integration with Payment Systems: Incorporating secure payment gateways for processing payments, refunds, and invoicing would streamline the reservation process.
- 2.Mobile App Development: Expanding the system to include a mobile interface would allow users to make and manage reservations from smartphones, increasing accessibility.
- 3.AI and Machine Learning: Implementing AI-driven features like dynamic pricing,

personalized recommendations, or predictive room availability could enhance the user experience and optimize revenue management.

4.Cloud Integration: Moving the system to the cloud would provide better scalability, data security, and remote access for both administrators and users.

5.Event Management Expansion: Adding advanced event management features like scheduling, attendee tracking, and resource allocation would make the system more versatile for hotels or venues hosting large events.

6.Analytics and Reporting: Incorporating detailed reporting and analytics features to track occupancy rates, booking trends, and customer preferences could provide valuable insights for decision-making.

REFERENCES

Books on System Design:

- **Designing Data-Intensive Applications** by Martin Kleppmann
Focuses on data modeling and system architecture, which are crucial when designing reservation systems that handle user interactions and booking data.
- **Patterns of Enterprise Application Architecture** by Martin Fowler
Discusses architectural patterns for enterprise systems, which can be helpful in structuring hotel reservation systems.

Online Articles:

- **Hotel Reservation System: Architecture and Design**
Articles like this explore how to design the system architecture for a hotel booking platform, considering aspects such as user authentication, payment integration, and real-time availability updates.

Example: <https://dilfuruzmukhtar.medium.com/hotel-booking-system-design-architecture-848e8bfa58da>

- **Building a Simple Hotel Reservation System (Medium)**
This covers basic implementation of a reservation system using technologies like Node.js, Express, and MongoDB.

Software Design Patterns and Approaches:

- **Model-View-Controller (MVC):** This is a common design pattern used for hotel reservation systems to separate the data model (hotel rooms, bookings) from the user interface.
- **State Management:** Handling states like "available," "booked," "pending" for rooms, and managing their transitions can be done effectively with state design patterns.

APPENDIX A

(SOURCE CODE)

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

public class HotelReservationSystem extends Frame implements ActionListener {
    // Components for the GUI
    Label nameLabel, roomTypeLabel, daysLabel, outputLabel;
    TextField nameField, daysField;
    Choice roomTypeChoice;
    Button submitButton, clearButton, showRoomsButton;
    TextArea outputArea;

    // Room availability
    int totalRooms = 18; // Total initial rooms
    int singleRooms = 10; // Initial count of Single rooms
    int doubleRooms = 5; // Initial count of Double rooms
    int suiteRooms = 3; // Initial count of Suite rooms

    // Constructor to set up the GUI
    public HotelReservationSystem() {
        setTitle("Hotel Reservation System");
        setSize(500, 400);
        setLayout(new FlowLayout());
        setBackground(Color.LIGHT_GRAY);

        // Name Label and Field
        nameLabel = new Label("Name:");
        nameField = new TextField(30);
```

```

add(nameLabel);
add(nameField);

// Room Type Label and Choice
roomTypeLabel = new Label("Room Type:");
roomTypeChoice = new Choice();
updateRoomTypeChoice(); // Dynamically update the dropdown
add(roomTypeLabel);
add(roomTypeChoice);

// Number of Days Label and Field
daysLabel = new Label("Number of Days:");
daysField = new TextField(5);
add(daysLabel);
add(daysField);

// Submit and Clear Buttons
submitButton = new Button("Submit");
clearButton = new Button("Clear");
showRoomsButton = new Button("Show Number of Rooms");
add(submitButton);
add(clearButton);
add(showRoomsButton);

// Output Area
outputLabel = new Label("Reservation Details:");
outputArea = new TextArea(5, 40);
outputArea.setEditable(false);
add(outputLabel);

```

```

add(outputArea);

// Adding Action Listeners
submitButton.addActionListener(this);
clearButton.addActionListener(this);
showRoomsButton.addActionListener(this);

// Window Listener for closing the application
addWindowListener(new WindowAdapter() {
    public void windowClosing(WindowEvent we) {
        dispose();
    }
});

setVisible(true);
}

// Update room type dropdown with availability
private void updateRoomTypeChoice() {
    roomTypeChoice.removeAll(); // Clear existing choices
    roomTypeChoice.add("Single (" + singleRooms + " available)");
    roomTypeChoice.add("Double (" + doubleRooms + " available)");
    roomTypeChoice.add("Suite (" + suiteRooms + " available)");
}

// Handle button actions
public void actionPerformed(ActionEvent e) {
    if (e.getSource() == submitButton) {
        String name = nameField.getText();

```

```

String selectedRoom = roomTypeChoice.getSelectedItemAt();
String daysText = daysField.getText();

if (name.isEmpty() || daysText.isEmpty()) {
    outputArea.setText("Please fill in all fields.");
    return;
}

try {
    int days = Integer.parseInt(daysText);
    double rate = 0;

    if (selectedRoom.startsWith("Single") && singleRooms > 0) {
        rate = 100;
        singleRooms--;
        totalRooms--;
    } else if (selectedRoom.startsWith("Double") && doubleRooms > 0) {
        rate = 150;
        doubleRooms--;
        totalRooms--;
    } else if (selectedRoom.startsWith("Suite") && suiteRooms > 0) {
        rate = 300;
        suiteRooms--;
        totalRooms--;
    } else {
        outputArea.setText("Sorry, no rooms available for the selected type.");
        return;
    }
}

```

```

double totalCost = days * rate;
outputArea.setText("Reservation Successful!\n" +
    "Name: " + name + "\n" +
    "Room Type: " + selectedRoom.split(" ")[0] + "\n" +
    "Number of Days: " + days + "\n" +
    "Total Cost: $" + totalCost + "\n\n" +
    "Updated Room Availability:\n" +
    "Total Rooms: " + totalRooms + "\n" +
    "Single Rooms: " + singleRooms + "\n" +
    "Double Rooms: " + doubleRooms + "\n" +
    "Suite Rooms: " + suiteRooms);

// Update dropdown with new availability
updateRoomTypeChoice();
} catch (NumberFormatException ex) {
    outputArea.setText("Please enter a valid number for days.");
}
} else if (e.getSource() == clearButton) {
    nameField.setText("");
    daysField.setText("");
    roomTypeChoice.select(0);
    outputArea.setText("Fields cleared.");
} else if (e.getSource() == showRoomsButton) {
    outputArea.setText("Current Room Availability:\n" +
        "Total Rooms: " + totalRooms + "\n" +
        "Single Rooms: " + singleRooms + "\n" +
        "Double Rooms: " + doubleRooms + "\n" +
        "Suite Rooms: " + suiteRooms);
}

```

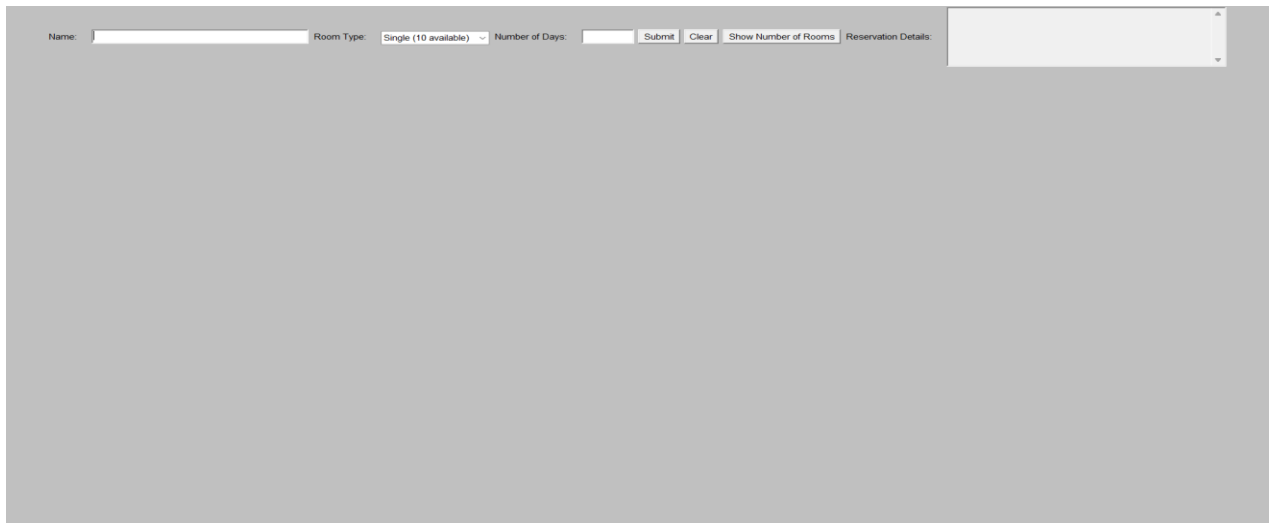


```
}

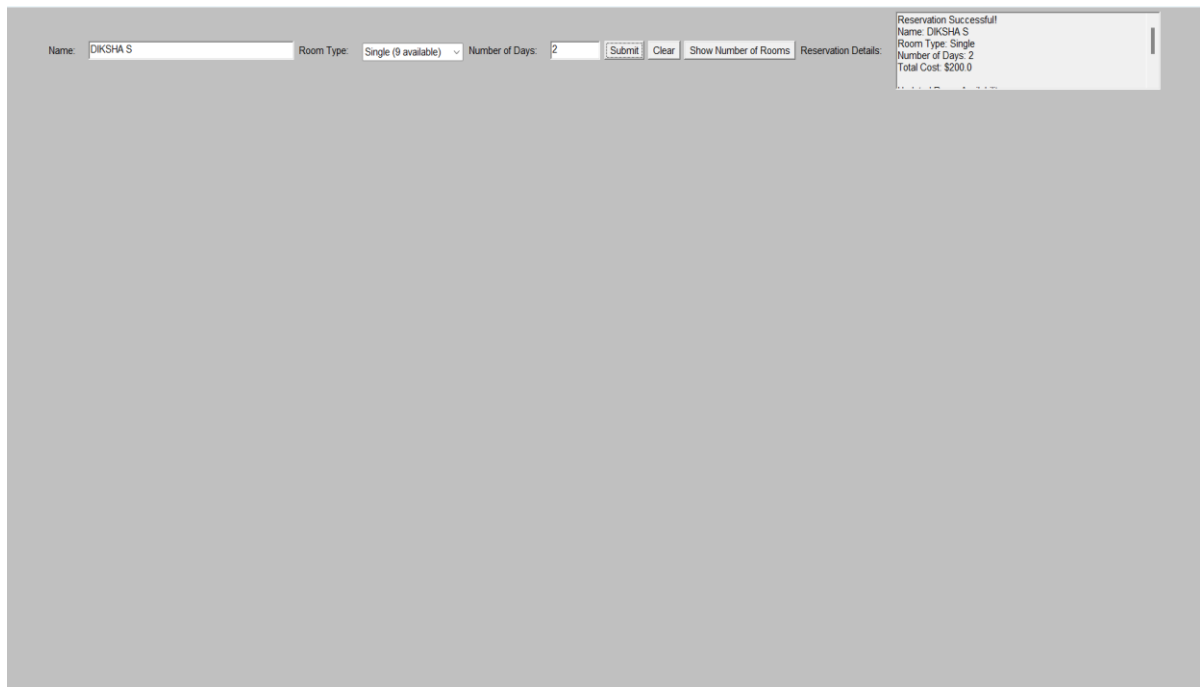
// Main method to launch the application
public static void main(String[] args) {
    new HotelReservationSystem();
}
}
```

APPENDIX B

(SCREENSHOTS)



A screenshot of a web application's reservation form. The form is set against a light gray background. At the top, there is a header bar containing several input fields and buttons. The 'Name' field is empty. The 'Room Type' dropdown menu is set to 'Single (10 available)'. The 'Number of Days' field is empty. To the right of these fields are buttons for 'Submit', 'Clear', and 'Show Number of Rooms'. Further right is a 'Reservation Details' section, which is currently empty. The main body of the page is a large, empty gray rectangle.



A screenshot of the same reservation form, but now showing a successful reservation. The 'Name' field is filled with 'DIKSHA S'. The 'Room Type' dropdown menu is set to 'Single (9 available)'. The 'Number of Days' field is filled with '2'. The 'Submit' button is highlighted with a dashed border. The 'Reservation Details' section on the right now displays the following information: 'Reservation Successful', 'Name: DIKSHA S', 'Room Type: Single', 'Number of Days: 2', and 'Total Cost: \$200.0'. The main body of the page remains a large, empty gray rectangle.

Name:

Room Type:

Number of Days:

Reservation Details:

Updated Room Availability:
Total Rooms: 17
Single Rooms: 9
Double Rooms: 5
Suite Rooms: 3