

HOTEL MANAGEMENT SYSTEM IN JAVA

MINOR PROJECT REPORT

By

**AVANI BADERIYA (RA2211003011780)
NEHA MAURYA (RA2211003011787)
MARUSHKA SINGHANIA (RA2211003011810)**

Under the guidance of

MS.M HEMA

In partial fulfilment for the Course

of

21CSC203P – ADVANCED PROGRAMMING PRACTICE

in **Computing Technologies**



FACULTY OF ENGINEERING AND TECHNOLOGY

SCHOOL OF COMPUTING

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR

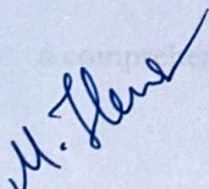
NOVEMBER 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

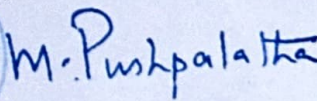
BONAFIDE CERTIFICATE

Certified that this minor project report for the course **21CSC203P ADVANCED PROGRAMMING PRACTICE** entitled in **Hotel Management System** is the bonafide work of **AVANI BADERIYA (RA2211003011780), NEHA MAURYA (RA2211003011787), and MARUSHKA SINGHANIA (RA2211003011810)** who carried out the work under my supervision.


SIGNATURE

Ms. M. HEMA
Assistant Professor
Computing Technologies Department
SRM Institute of Science and Technology
Kattankulathur




SIGNATURE

Dr. M. PUSHPALATHA
Professor and Head
Computing Technologies Department
SRM Institute of Science and Technology
Kattankulathur

ABSTRACT

This tool is designed for hotel management, helping handle tasks like storing customer information, room bookings of various types, food orders for specific rooms, room cancellations, and generating bills. It operates through a user-friendly menu, making it easy to navigate. The program continues to run until the user decides to exit, providing a seamless and efficient management experience. With a focus on simplicity, the tool streamlines hotel activities, ensuring smooth customer interactions, room reservations, and food services. Its user-friendly design allows staff to manage tasks effortlessly, enhancing overall operational efficiency. From booking different room types to handling food orders and generating bills, this tool is a comprehensive solution for effective hotel management.

.

ACKNOWLEDGEMENT

We express our heartfelt thanks to our honorable **Vice Chancellor Dr. C. MUTHAMIZHCHELVAN**, for being the beacon in all our endeavors.

We would like to express my warmth of gratitude to our **Registrar Dr. S. Ponnusamy**, for his encouragement.

We express our profound gratitude to our **Dean (College of Engineering and Technology) Dr. T. V.Gopal**, for bringing out novelty in all executions.

We would like to express my heartfelt thanks to Chairperson, School of Computing **Dr. Revathi Venkataraman**, for imparting confidence to complete my course project

We wish to express my sincere thanks to **Course Audit Professors Dr. Vadivu. G , Professor, Department of Data Science and Business Systems and Dr. Sasikala. E Professor, Department of Data Science and Business Systems and Course Coordinators** for their constant encouragement and support.

We are highly thankful to our my Course project Faculty **Ms.M Hema , Assistant Professor , Computing Technologies** for her assistance, timely suggestion and guidance throughout the duration of this course project.

Finally, we thank our parents and friends near and dear ones who directly and indirectly contributed to the successful completion of our project. Above all, I thank the almighty for showering his blessings on me to complete my Course project.

TABLE OF CONTENTS

CHAPTER NO	CONTENTS	PAGE NO
1	INTRODUCTION	5
2	LITERATURE SURVEY	6
3	DATABASE DESIGN	7
4	PROGRAM	8
5	RESULT/OUTPUT	22
6	CONCLUSION	26
7	REFERENCES	27

1. INTRODUCTION

Welcome to our innovative Hotel Management Tool, a dynamic solution crafted to streamline and enhance the efficiency of hotel operations. Developed using Java Swing applets and other components, this tool transforms the management of hotel activities into a seamless and user-friendly experience.

Our Java Swing-based application empowers hotel staff with a menu-driven program, facilitating the storage of customer details, diverse room bookings, food ordering for specific rooms, room cancellations, and bill generation. The intuitive design of the tool ensures that users can effortlessly navigate through various functionalities, making it an ideal choice for hotel management.

The use of Java Swing applets adds a layer of sophistication to the user interface, offering a visually appealing and interactive experience. The application runs continuously until the user decides to exit, providing a persistent and convenient platform for managing hotel tasks. Whether it's accommodating different room types, handling food orders, or generating accurate bills, our Hotel Management Tool stands as a comprehensive solution, combining the power of Java Swing applets with efficient management components. Join us on this journey to revolutionize hotel management with a tool that not only simplifies tasks but also adds a touch of elegance to the user experiences.

2. LITERATURE SURVEY

According to Kalaskar P. (2013), hotels are always classified based on the amenities they offer, such as 5-star hotels indicating luxury. Hotels with fewer facilities and poor services receive lower star ratings. These ratings are determined by evaluating all the factors necessary for providing the best facilities. The assessment is done every five years.

According to Popat K. (2013), running a hotel system needs a brave team that coordinates well with all the facilities and services provided in the hotel. All aspects of managing the system, like laundry service, food and catering, finance department, and hotel administration, should function smoothly with no defaults.

According to Saraswathy R. and Premakumari P. (2014) promoting hotels is crucial to attract customer interest. In today's world, where technical facilities like the internet, smartphones, computers, and laptops are widespread, online promotion becomes vital. The youth, deeply engaged in these facilities, makes online promotion effective in attracting customers. Over the last two decades, online tourism, also known as e-marketing, has become a leading marketing strategy.

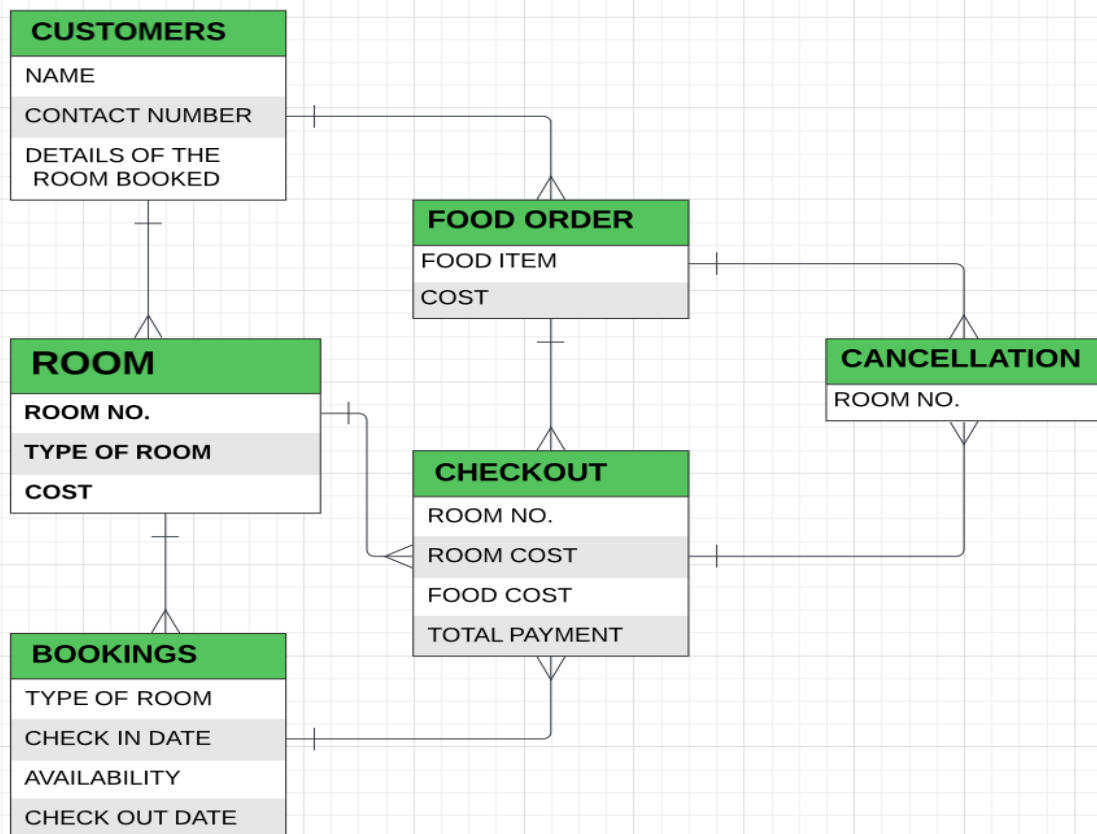
According to Sharma A. and Kukreja S. (2013), the hotel industry is a significant part of tourist traffic, although India is lagging behind other countries. This is attributed to poor facilities, substandard food, and customer dissatisfaction in some hotels.

According to Batra M. (2014) the factors related to hotel services are not meeting customer satisfaction. It's crucial to thoroughly analyze these factors. Some luxury hotels charge excessively high prices compared to their services, negatively impacting customer feedback and

satisfaction.

3. DATABASE DESIGN

HOTEL MANAGEMENT DATABASE



4. PROGRAM

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.List;

class Guest {
    private String name;
    private String contact;

    public Guest(String name, String contact) {
        this.name = name;
        this.contact = contact;
    }

    public String getName() {
        return name;
    }

    public String getContact() {
        return contact;
    }

    @Override
    public String toString() {
```

```
        return "Name: " + name + "\nContact: " + contact;
    }

}
```

```
class Room {
    private int roomNumber;
    private boolean isOccupied;
    private List<String> orderedItems;
    private Guest guest;
    private double roomCost;

    public Room(int roomNumber, double roomCost) {
        this.roomNumber = roomNumber;
        this.isOccupied = false;
        this.orderedItems = new ArrayList<>();
        this.guest = null;
        this.roomCost = roomCost;
    }

    public int getRoomNumber() {
        return roomNumber;
    }

    public boolean isOccupied() {
        return isOccupied;
    }
}
```

```

public void setOccupied(boolean occupied) {
    isOccupied = occupied;
    if (!occupied) {
        // Clear ordered items and guest details when the room is unoccupied
        orderedItems.clear();
        guest = null; }
}

public List<String> getOrderedItems() {
    return orderedItems;
}

public void addOrderedItem(String item) {
    orderedItems.add(item);
}

public double calculateTotalCost() {
    // Calculate the total cost including the room cost and the cost of ordered
items
    double itemsCost = orderedItems.size() * 10.0; // Example cost for ordered
items
    return roomCost + itemsCost;
}

public Guest getGuest() {
    return guest;
}

```

```

    public void setGuest(Guest guest) {
        this.guest = guest;
    }
}

class HotelManagementSystem {
    private List<Room> rooms;

    public HotelManagementSystem() {
        rooms = new ArrayList<>();
        initializeRooms();
    }

    private void initializeRooms() {
        // Different types of rooms with their respective costs
        rooms.add(new Room(101, 100.0));
        rooms.add(new Room(102, 120.0));
        rooms.add(new Room(103, 150.0));
        rooms.add(new Room(104, 200.0));
        rooms.add(new Room(105, 250.0));
        rooms.add(new Room(106, 300.0));
        rooms.add(new Room(107, 180.0));
        rooms.add(new Room(108, 220.0));
        rooms.add(new Room(109, 270.0));
        rooms.add(new Room(110, 320.0));
    }

    public List<Room> getRooms() {

```

```

        return rooms;
    }

    public Room getRoomByNumber(int roomNumber) {
        for (Room room : rooms) {
            if (room.getRoomNumber() == roomNumber) {
                return room;
            }
        }
        return null;
    }
}

public class HotelManagementGUI {
    private HotelManagementSystem hotelManagementSystem;

    private JFrame frame;
    private JButton viewRoomsButton;
    private JButton bookRoomButton;
    private JButton orderFoodButton;
    private JButton cancelBookingButton;

    public HotelManagementGUI() {
        hotelManagementSystem = new HotelManagementSystem();

        frame = new JFrame("Hotel Management System");
    }
}

```

```
frame.setSize(400, 200);  
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
```

```
viewRoomsButton = new JButton("View Rooms");  
viewRoomsButton.addActionListener(new ActionListener() {  
    @Override  
    public void actionPerformed(ActionEvent e) {  
        showRooms();  
    }  
});
```

```
bookRoomButton = new JButton("Book Room");  
bookRoomButton.addActionListener(new ActionListener() {  
    @Override  
    public void actionPerformed(ActionEvent e) {  
        bookRoom();  
    }  
});
```

```
orderFoodButton = new JButton("Order Food");
```

```
orderFoodButton.addActionListener(new ActionListener() {  
    @Override  
    public void actionPerformed(ActionEvent e) {  
        orderFood();  
    }  
});
```

```
cancelBookingButton = new JButton("Cancel Booking");
cancelBookingButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        cancelBooking();
    }
});
```

```
JPanel panel = new JPanel();
panel.add(viewRoomsButton);
panel.add(bookRoomButton);
panel.add(orderFoodButton);
panel.add(cancelBookingButton);
```

```
frame.add(panel);
frame.setVisible(true);
}
```

```
private void showRooms() {
    StringBuilder message = new StringBuilder("Room List:\n");
    for (Room room : hotelManagementSystem.getRooms()) {
        message.append("Room ").append(room.getRoomNumber()).append(":
");
        if (room.isOccupied()) {

            message.append("Occupied\n");
            message.append(room.getGuest()).append("\n");
        }
    }
}
```

```

        } else {
            message.append("Available\n");
        }
        message.append("Cost:
$").append(room.calculateTotalCost()).append("\n\n");
    }
    JOptionPane.showMessageDialog(frame, message.toString(), "Room
Information", JOptionPane.INFORMATION_MESSAGE);
}

private void bookRoom() {
    try {
        String[] roomOptions = {"101 - Standard ($100)", "102 - Deluxe
($120)", "103 - Suite ($150)",
            "104 - Executive Suite ($200)", "105 - Presidential Suite ($250)",
"106 - Penthouse ($300)",
            "107 - Junior Suite ($180)", "108 - Family Suite ($220)", "109 -
Royal Suite ($270)",
            "110 - Grand Presidential Suite ($320)"};

        String selectedRoom = (String) JOptionPane.showInputDialog(frame,
"Select Room Type", "Room Selection",
JOptionPane.QUESTION_MESSAGE, null, roomOptions, roomOptions[0]);

        int roomNumber = Integer.parseInt(selectedRoom.substring(0,
3).trim());
        Room room =
hotelManagementSystem.getRoomByNumber(roomNumber);

```



```

        if (room != null) {
            if (!room.isOccupied()) {

                String guestName = JOptionPane.showInputDialog(frame, "Enter
Guest Name:");

                String guestContact = JOptionPane.showInputDialog(frame, "Enter
Guest Contact:");

                Guest guest = new Guest(guestName, guestContact);
                room.setGuest(guest);
                room.setOccupied(true);

                JOptionPane.showMessageDialog(frame, "Room " + roomNumber
+ " booked successfully!\n" + guest + "\nCost: $" + room.calculateTotalCost(),
"Success", JOptionPane.INFORMATION_MESSAGE);
            } else {
                JOptionPane.showMessageDialog(frame, "Room " + roomNumber
+ " is already occupied.", "Error", JOptionPane.ERROR_MESSAGE);
            }
        } else {
            JOptionPane.showMessageDialog(frame, "Invalid room number.",
"Error", JOptionPane.ERROR_MESSAGE);
        }
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(frame, "Invalid input. Please enter a
valid room number.", "Error", JOptionPane.ERROR_MESSAGE);
    }
}

```

```

private void orderFood() {
    String roomNumberString = JOptionPane.showInputDialog(frame, "Enter
Room Number to Order Food:");
    try {
        int roomNumber = Integer.parseInt(roomNumberString);
        Room room =
hotelManagementSystem.getRoomByNumber(roomNumber);

        if (room != null) {
            if (room.isOccupied()) {
                showFoodMenu(room);
            } else {
                JOptionPane.showMessageDialog(frame, "Room " + roomNumber
+ " is not occupied. Please book the room first.", "Error",
JOptionPane.ERROR_MESSAGE);
            }
        } else {
            JOptionPane.showMessageDialog(frame, "Invalid room number.",
"Error", JOptionPane.ERROR_MESSAGE);
        }
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(frame, "Invalid input. Please enter a
valid room number.", "Error", JOptionPane.ERROR_MESSAGE);
    }
}

```

```

private void cancelBooking() {

```

```

String roomNumberString = JOptionPane.showInputDialog(frame, "Enter
Room Number to Cancel Booking:");

try {
    int roomNumber = Integer.parseInt(roomNumberString);
    Room room =
hotelManagementSystem.getRoomByNumber(roomNumber);
    if (room != null) {
        if (room.isOccupied()) {
            room.setOccupied(false);
            JOptionPane.showMessageDialog(frame, "Booking for Room " +
roomNumber + " canceled successfully!", "Success",
JOptionPane.INFORMATION_MESSAGE);

        } else {
            JOptionPane.showMessageDialog(frame, "Room " + roomNumber
+ " is not occupied. No booking to cancel.", "Error",
JOptionPane.ERROR_MESSAGE);
        }
    } else {
        JOptionPane.showMessageDialog(frame, "Invalid room number.",
"Error", JOptionPane.ERROR_MESSAGE);
    }
} catch (NumberFormatException ex) {
    JOptionPane.showMessageDialog(frame, "Invalid input. Please enter a
valid room number.", "Error", JOptionPane.ERROR_MESSAGE);
}
}

```

```

private void showFoodMenu(Room room) {
    String[] menuItems = {"Burger", "Pizza", "Pasta", "Salad"};
    double[] menuPrices = {10.0, 12.0, 8.0, 6.0};
    boolean[] selectedItems = new boolean[menuItems.length];

    int result = JOptionPane.showConfirmDialog(frame,
getMenuOptions(menuItems, menuPrices, selectedItems), "Select Food Items",
JOptionPane.OK_CANCEL_OPTION, JOptionPane.QUESTION_MESSAGE);

    if (result == JOptionPane.OK_OPTION) {
        double totalCost = 0.0;
        for (int i = 0; i < menuItems.length; i++) {
            if (selectedItems[i]) {
                room.addOrderedItem(menuItems[i]);
                totalCost += menuPrices[i];
            }
        }
        DecimalFormat df = new DecimalFormat("#.##");
        JOptionPane.showMessageDialog(frame, "Food ordered
successfully!\nTotal Food Cost: $" + df.format(totalCost), "Success",
JOptionPane.INFORMATION_MESSAGE);
    }
}

```

```

private Object[] getMenuOptions(String[] menuItems, double[] menuPrices,

```

```

boolean[] selectedItems) {
    Object[] options = new Object[menuItems.length * 2 + 1];
    for (int i = 0; i < menuItems.length; i++) {
        options[i * 2] = menuItems[i];
        options[i * 2 + 1] = "$" + menuPrices[i];
    }
    options[menuItems.length * 2] = "Total Cost: $0.00"; // Placeholder for
total cost

    JCheckBox[] checkboxes = new JCheckBox[menuItems.length];
    for (int i = 0; i < menuItems.length; i++) {
        checkboxes[i] = new JCheckBox(menuItems[i]);
    }

    options[menuItems.length * 2] = checkboxes;

    int result = JOptionPane.showConfirmDialog(frame, options, "Select Food
Items", JOptionPane.OK_CANCEL_OPTION,
JOptionPane.QUESTION_MESSAGE);

    if (result == JOptionPane.OK_OPTION) {

        double totalCost = 0.0;
        for (int i = 0; i < menuItems.length; i++) {
            if (checkboxes[i].isSelected()) {
                selectedItems[i] = true;
                totalCost += menuPrices[i];
            }
        }
    }
}

```

```

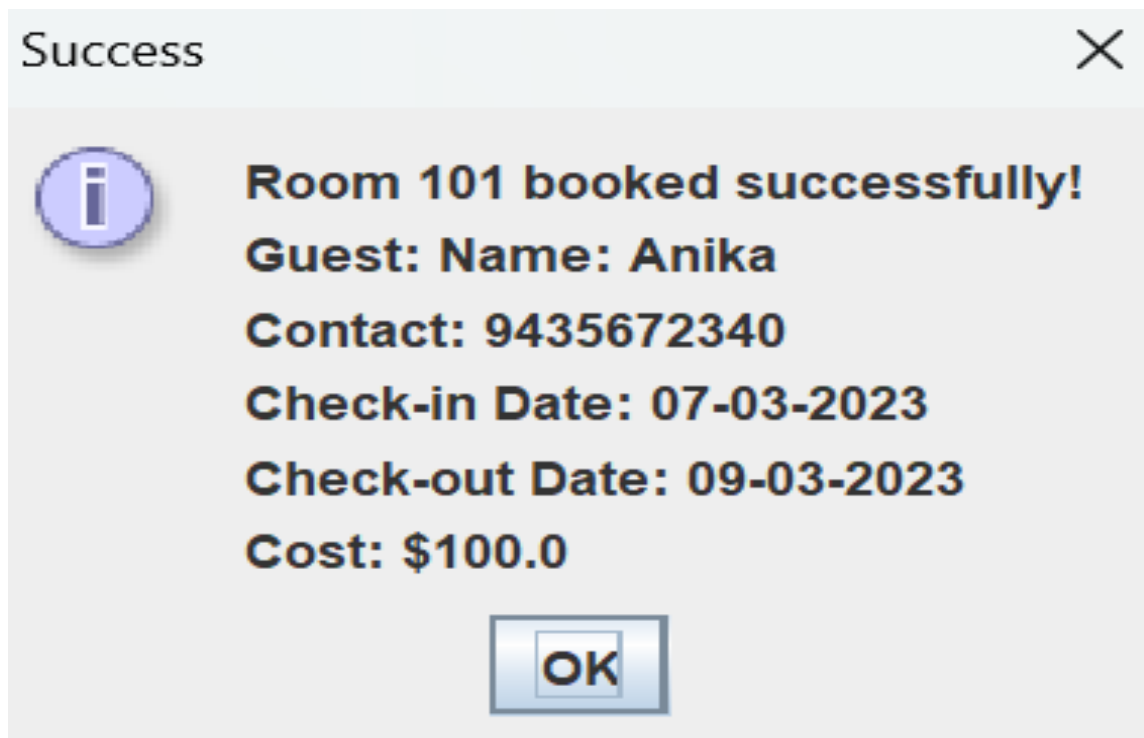
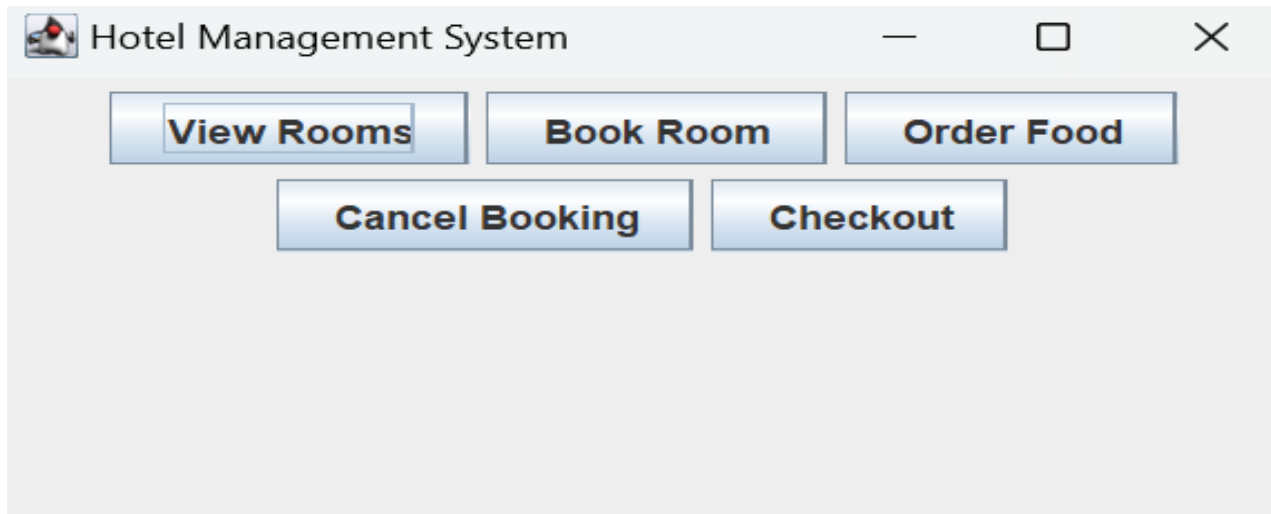
        }
    }
    DecimalFormat df = new DecimalFormat("#.##");
    options[menuItems.length * 2] = "Total Cost: $" + df.format(totalCost);
}

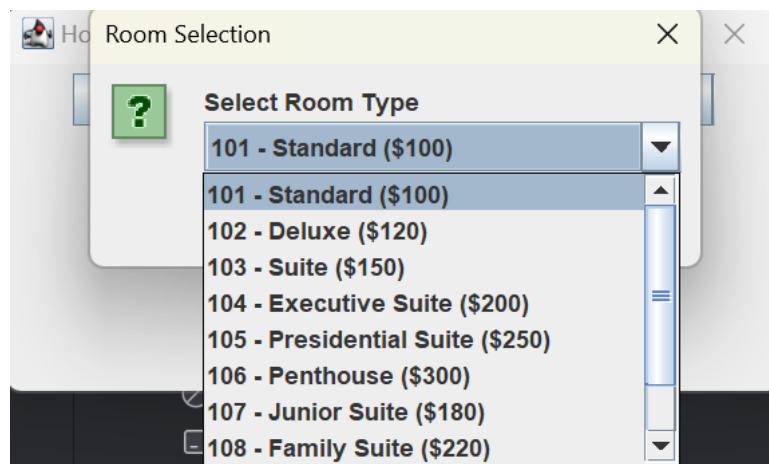
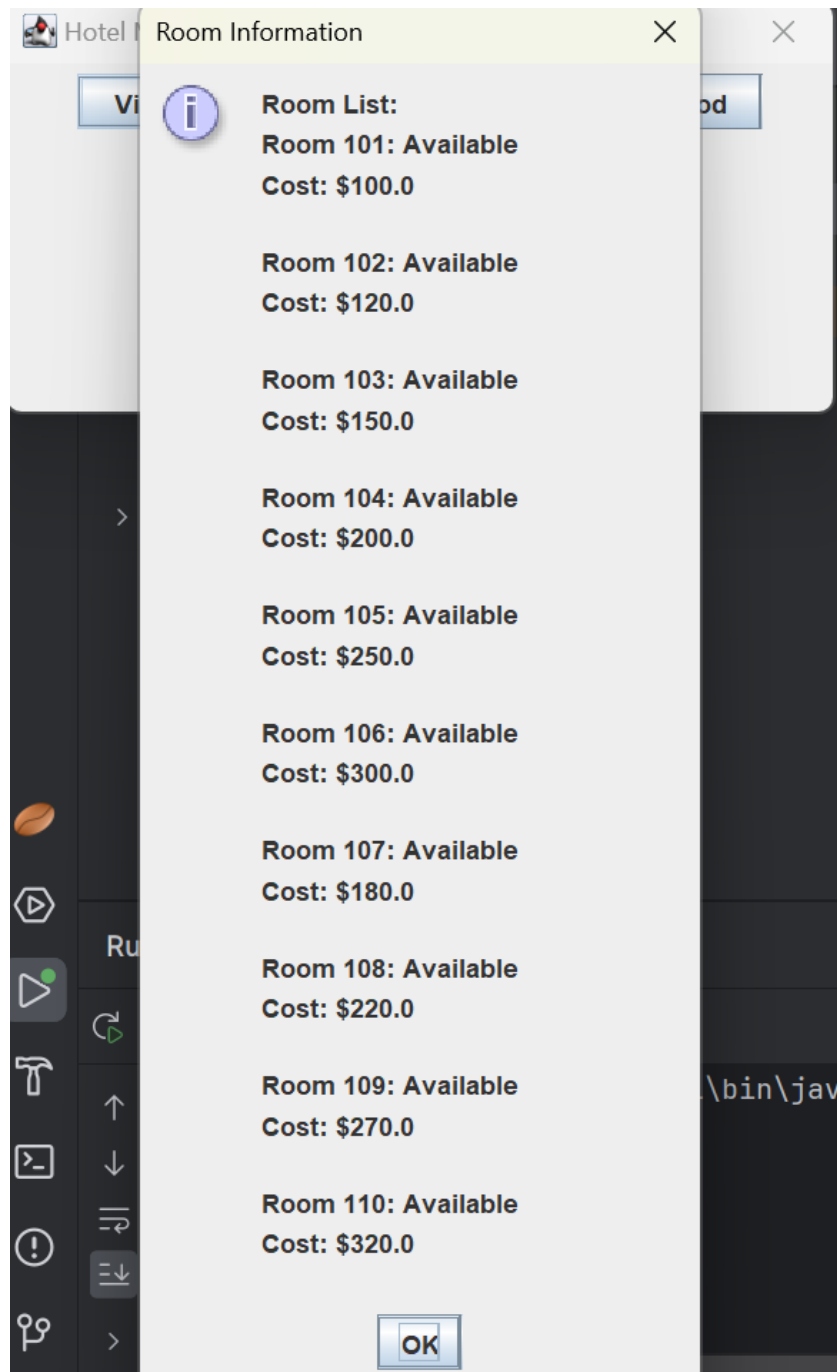
return options;
}

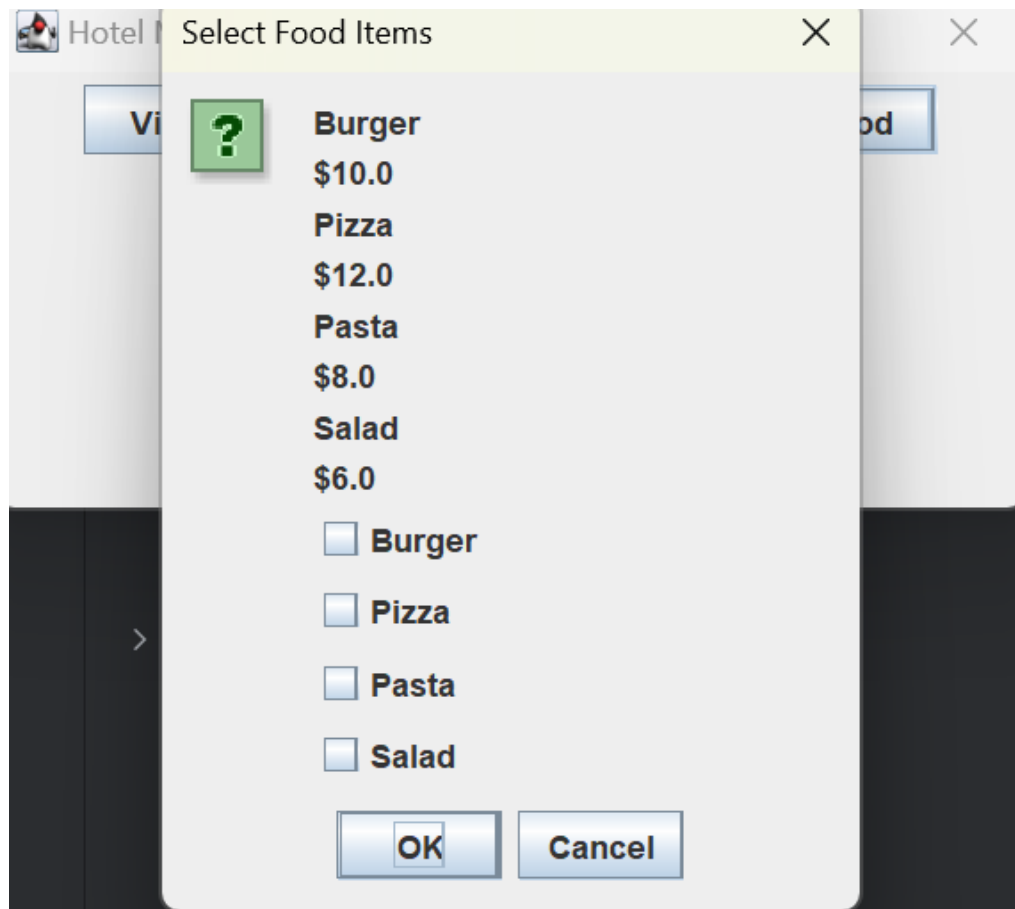
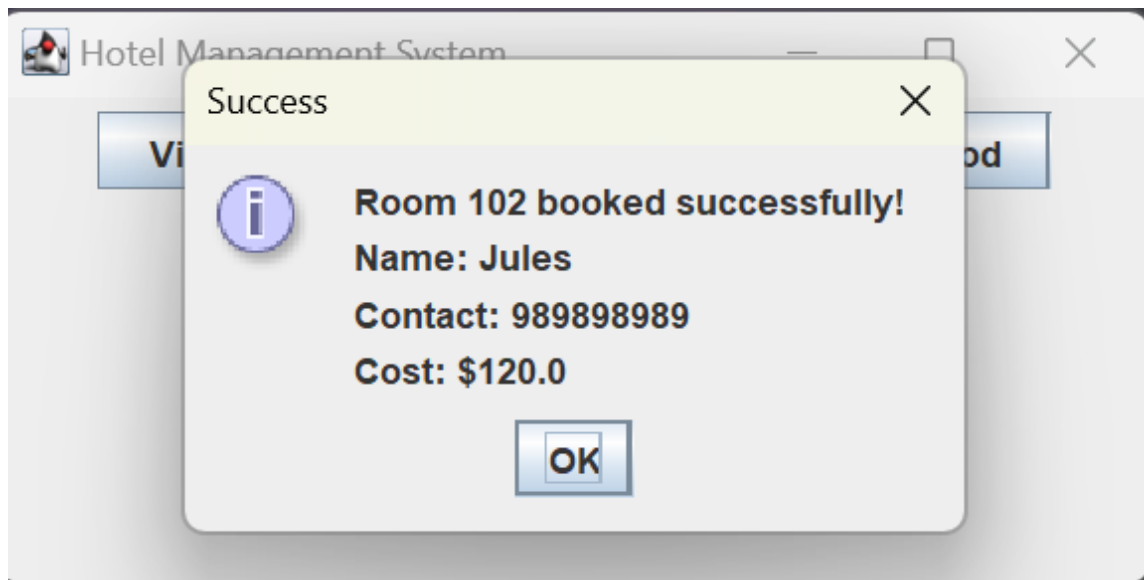
public static void main(String[] args) {
    SwingUtilities.invokeLater(new Runnable() {
        @Override
        public void run() {
            new HotelManagementGUI();
        }
    });
}
}

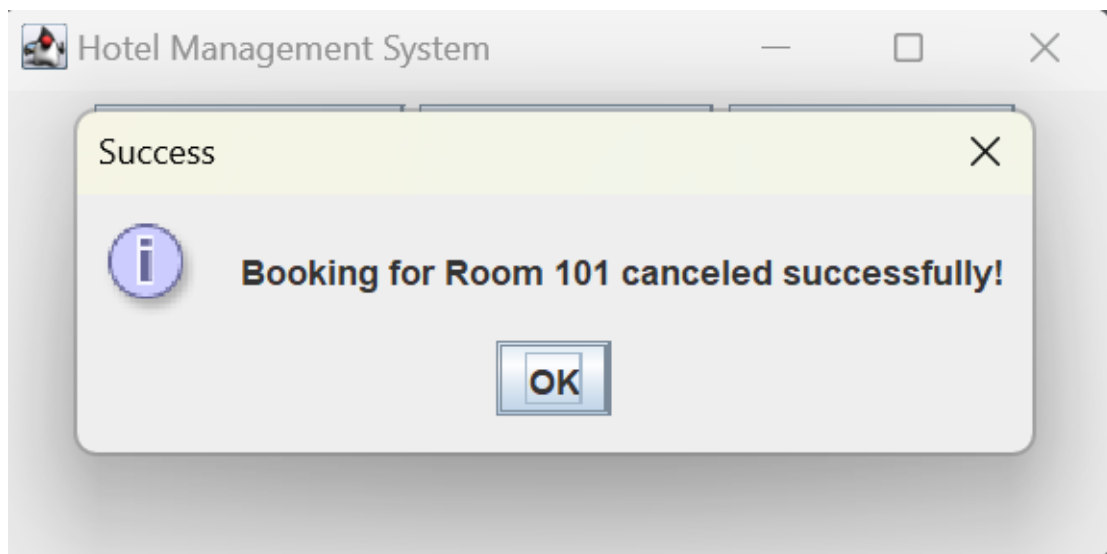
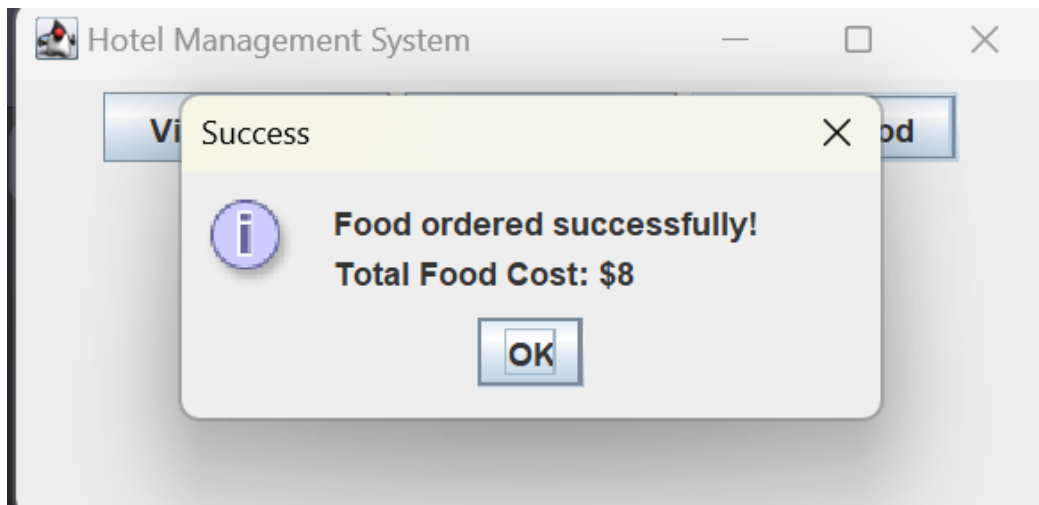
```

5. RESULTS/OUTPUT









6. CONCLUSION

In conclusion, our Hotel Management Tool, powered by Java Swing applets, is a user-friendly solution for the hospitality industry. It efficiently handles tasks like customer details, room bookings, food orders, and bill generation through a simple menu. The use of Java Swing applets enhances the visual and interactive aspects of the tool.

This project goes beyond just managing hotels; it reflects our dedication to improving operations and customer satisfaction. The tool's continuous runtime ensures constant accessibility, and its comprehensive functionality meets diverse hotel needs. As we conclude, this tool signifies a significant step in modernizing and simplifying hotel operations, showcasing how technology can transform and optimize tasks in the ever-changing hospitality landscape.

7. REFERENCES

- <https://www.geeksforgeeks.org/hotel-management-system/>
- <https://nevonprojects.com/hotel-management-system/>
- <https://www.mifratech.com/public/blog-page/Hotel+Management+System>