Hotel Reservation System

Introduction:

The Hotel Reservation System is a console-based application built in Java. It supports search functionality for available hotel rooms, booking hotel rooms, and transaction handling for paying. The project provides a mock representation of a hotel management system's fundamental processes. This is a showcase for applying the concepts of object-oriented programming in practical problem-solving.

Objective:

a. Hotel Module

- It models the hotel name and location, along with a collection of rooms.
- Allows adding rooms of certain types and searching available rooms by type.

b. Room Module

- Each room will have specific attributes to it, such as the room number, room type, availability, and price.
- Enable to book a room as available or unavailable

c. Reservation Module

- Details of a booking: the hotel, the room, the guest name, the date of check-in.
- Acts as a record of every confirmed reservation.

d. Payment Module

- It conducts the payment process for a reservation.
- You can use the net banking gateway which comes with login feature, credit card, UPI, PhonePe.
- Payments that are successful for invoice and payment bills.

e. Main Application (User Interface)

- This gives a working command-line interface to play around with the user.
- Search for rooms, book a room, and log out of the application.

Advantages of the System

- A very simple to user-friendly interface
- Thus promotes the learning of OOP in the real world.
- It can be redesigned and updated easily.

Program:

```
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.text.SimpleDateFormat;
import java.text.ParseException;
import java.util.Scanner;

// Hotel Class
class Hotel {
    private String name;
    private String location;
    private List<Room> rooms;

    public Hotel(String name, String location) {
        this.name = name;
        this.location = location;
        this.rooms = new ArrayList<>();
    }

    public String getName() {
        return name;
    }

    public String getLocation() {
```

```
        return location;
    }

    public List<Room> getRooms() {
        return rooms;
    }

    public void addRoom(Room room) {
        this.rooms.add(room);
    }

    public List<Room> searchAvailableRooms(String roomType) {
        List<Room> availableRooms = new ArrayList<>();
        for (Room room : rooms) {
            if (room.getRoomType().equalsIgnoreCase(roomType) &&
room.isAvailable()) {
                availableRooms.add(room);
            }
        }
        return availableRooms;
    }

    public Room findRoomByNumber(int roomNumber) {
        for (Room room : rooms) {
            if (room.getRoomNumber() == roomNumber) {
                return room;
```

```
            }
        }
        return null;
    }
}

// Room Class
class Room {
    private int roomNumber;
    private String roomType;
    private boolean isAvailable;
    private double price; // Price in INR

    public Room(int roomNumber, String roomType, double price) {
        this.roomNumber = roomNumber;
        this.roomType = roomType;
        this.isAvailable = true;
        this.price = price;
    }

    public int getRoomNumber() {
        return roomNumber;
    }

    public String getRoomType() {
```

```
        return roomType;
    }

    public boolean isAvailable() {
        return isAvailable;
    }

    public void setAvailable(boolean available) {
        isAvailable = available;
    }

    public double getPrice() {
        return price;
    }
}

// Reservation Class
class Reservation {
    private Hotel hotel;
    private Room room;
    private String guestName;
    private Date checkInDate;

    public Reservation(Hotel hotel, Room room, String guestName, Date
checkInDate) {
        this.hotel = hotel;
```

```
        this.room = room;
        this.guestName = guestName;
        this.checkInDate = checkInDate;
    }

    public Hotel getHotel() {
        return hotel;
    }

    public Room getRoom() {
        return room;
    }

    public String getGuestName() {
        return guestName;
    }

    public Date getCheckInDate() {
        return checkInDate;
    }
}

// Payment Class
class Payment {
    private Reservation reservation;
```

```
    private String paymentMethod;
    private double amount;
    private boolean isPaid;

    public Payment(Reservation reservation, String paymentMethod) {
        this.reservation = reservation;
        this.paymentMethod = paymentMethod;
        this.amount = reservation.getRoom().getPrice();
        this.isPaid = false;
    }

    public boolean processPayment() {
        // Simulate payment processing
        System.out.println("Processing payment of ₹" + String.format("%.2f",
amount) + " using " + paymentMethod);
        isPaid = true;
        return true; // Indicate successful payment
    }

    public boolean isPaid() {
        return isPaid;
    }

    public double getAmount() {
        return amount;
    }
```

```
    public String getPaymentMethod() {
        return paymentMethod;
    }
}

// Main Class (Execution)
public class HotelReservationSystem {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Initialize hotel
        Hotel hotel = new Hotel("StayEasy Hotel", "Sample Location");

        // Initialize rooms with prices in INR
        hotel.addRoom(new Room(101, "Single", 4167.00));
        hotel.addRoom(new Room(102, "Single", 4167.00));
        hotel.addRoom(new Room(201, "Double", 8334.00));
        hotel.addRoom(new Room(202, "Double", 8334.00));
        hotel.addRoom(new Room(301, "Suite", 16668.00));

        // Main application loop
        while (true) {
            System.out.println("\nHOTEL RESERVATION SYSTEM");
            System.out.println("1. Search Available Rooms");
```

```
        System.out.println("2. Make a Reservation");
        System.out.println("3. Exit");
        System.out.print("Enter your choice: ");

        int choice = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        switch (choice) {
            case 1:
                searchRooms(hotel, scanner);
                break;
            case 2:
                makeReservation(hotel, scanner);
                break;
            case 3:
                System.out.println("Exiting the system. Thank you!");
                scanner.close();
                return;
            default:
                System.out.println("Invalid choice. Please try again.");
        }
    }
}

// Method to search for available rooms
```

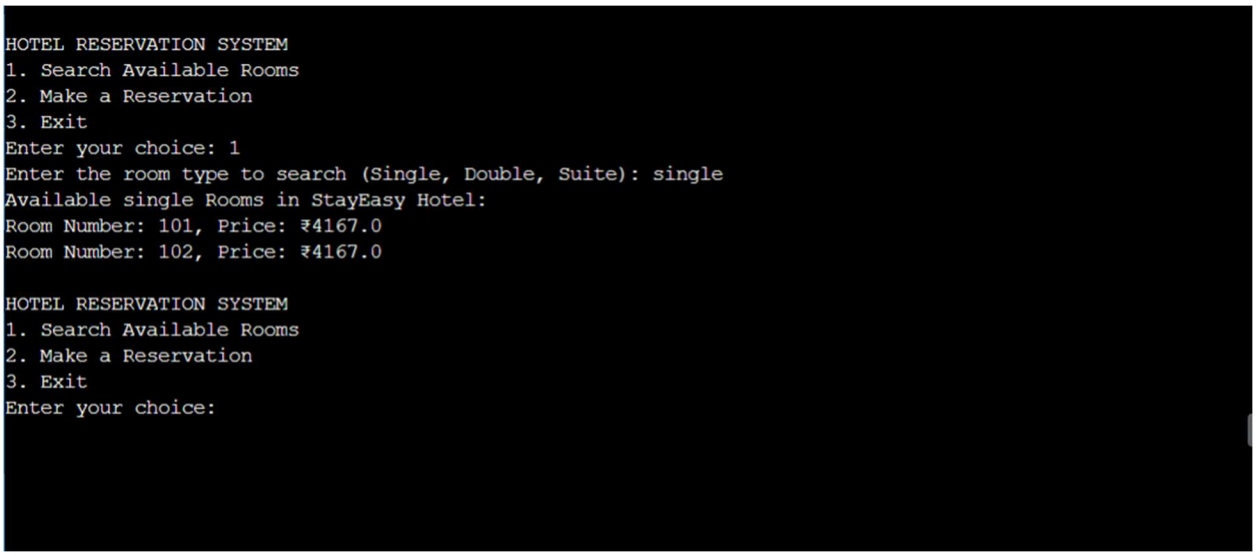
```
    private static void searchRooms(Hotel hotel, Scanner scanner) {
        System.out.print("Enter the room type to search (Single, Double, Suite): ");
        String roomType = scanner.nextLine();

        List<Room> availableRooms = hotel.searchAvailableRooms(roomType);
        System.out.println("Available " + roomType + " Rooms in " +
hotel.getName() + ":");
        if (availableRooms.isEmpty()) {
            System.out.println("No rooms available for the specified criteria.");
        } else {
            for (Room room : availableRooms) {
                System.out.println("Room Number: " + room.getRoomNumber() + ",
Price: ₹" + room.getPrice());
            }
        }
    }

    // Method to make a reservation
    private static void makeReservation(Hotel hotel, Scanner scanner) {
        System.out.print("Enter guest name: ");
        String guestName = scanner.nextLine();
        System.out.print("Enter the room number to reserve: ");
        int roomNumber = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        Room roomToReserve = hotel.findRoomByNumber(roomNumber);
```

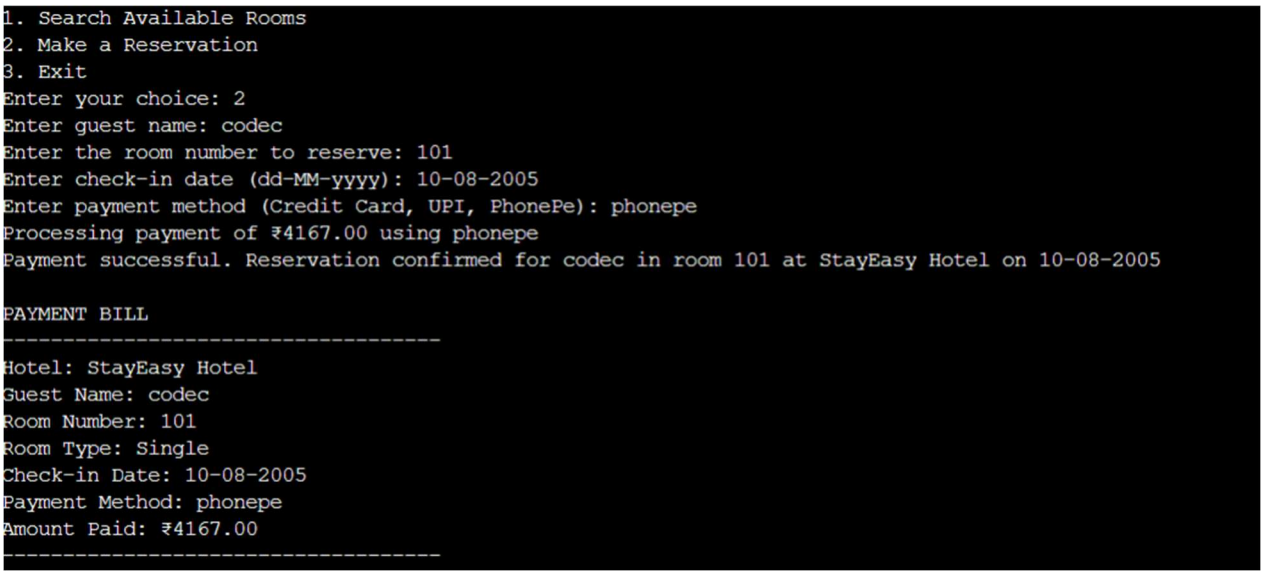
```
        if (roomToReserve == null || !roomToReserve.isAvailable()) {
            System.out.println("Room number " + roomNumber + " is not available. Please try again.");
            return;
        }

        SimpleDateFormat dateFormat = new SimpleDateFormat("dd-MM-yyyy");
        Date checkInDate = null;
        try {
            System.out.print("Enter check-in date (dd-MM-yyyy): ");
            String dateStr = scanner.nextLine();
            checkInDate = dateFormat.parse(dateStr);
        } catch (ParseException e) {
            System.out.println("Invalid date format. Please use dd-MM-yyyy.");
            return;
        }

        Reservation reservation = new Reservation(hotel, roomToReserve, guestName, checkInDate);

        // Process payment
        System.out.print("Enter payment method (Credit Card, UPI, PhonePe): ");
        String paymentMethod = scanner.nextLine();

        // Validate payment method (case-insensitive)
        if (!paymentMethod.equalsIgnoreCase("Credit Card") &&
```

```
            !paymentMethod.equalsIgnoreCase("UPI") &&
            !paymentMethod.equalsIgnoreCase("PhonePe")) {
        System.out.println("Invalid payment method. Please use Credit Card, UPI, or PhonePe.");
        return;
    }

    Payment payment = new Payment(reservation, paymentMethod);

    if (payment.processPayment()) {
        System.out.println("Payment successful. Reservation confirmed for " + reservation.getGuestName() +
                " in room " + reservation.getRoom().getRoomNumber() +
                " at " + reservation.getHotel().getName() +
                " on " + dateFormat.format(reservation.getCheckInDate()));
        roomToReserve.setAvailable(false); // Mark room as unavailable

        // Generate and display payment bill
        generatePaymentBill(payment, reservation, dateFormat);

    } else {
        System.out.println("Payment failed. Reservation not confirmed.");
    }
}

// Method to generate and display payment bill
```

```
    private static void generatePaymentBill(Payment payment, Reservation
reservation, SimpleDateFormat dateFormat) {
        System.out.println("\nPAYMENT BILL");
        System.out.println("-----------------------------------");
        System.out.println("Hotel: " + reservation.getHotel().getName());
        System.out.println("Guest Name: " + reservation.getGuestName());
        System.out.println("Room Number: " +
reservation.getRoom().getRoomNumber());
        System.out.println("Room Type: " + reservation.getRoom().getRoomType());
        System.out.println("Check-in Date: " +
dateFormat.format(reservation.getCheckInDate()));
        System.out.println("Payment Method: " + payment.getPaymentMethod());
        System.out.println("Amount Paid: ₹" + String.format("%.2f",
payment.getAmount()));
        System.out.println("-----------------------------------");
        System.out.println("Thank you for your reservation!");
    }
}
```

Output:





Conclusion

This project successfully implements the OOP concepts in a usable and modular booking system called the Hotel Reservation System. It addresses the important components of hotel management such as room listing, reservation management, and payment processing. This work is the first step towards more complex real-world applications.