AMRADIAMA: Hotel Management System Documentation

In Partial Fulfillment of the
Requirements for the ITEP 201
Object-Oriented Programming (OOP)

Dessa Marie Amparo

Karl Waitan Diamante

Paul Paolo Mamugay

RenzRamos

Project Title: AMRADIAMA: Hotel Management System

Date: [January 03, 2025]

Affiliation: Laguna State Polytechnic University

Submitted To: Mr. Nicko Albes

Course/Subject: Object-Oriented Programming (OOP)

Introduction

The ***AMRADIAMA: Hotel Management System*** is designed to make hotel operations simpler, faster, and more organized. It combines important tasks like taking reservations, managing guest details, tracking room availability, and handling promotions into one easy-to-use platform. By automating these tasks, the system reduces errors, saves time, and ensures that data is always accurate. It helps hotel staff work more efficiently and gives guests a smoother experience, from booking their stay to checking out.

This system is built to support the needs of small to medium-sized hotels, making it easier for them to keep up with guest expectations. Features like real-time updates on room availability, helpful reports, and secure online bookings ensure that operations run smoothly and reliably. With its simple design, the AMRADIAMA: Hotel Management System is easy for both staff and managers to use, while guests enjoy quicker and more convenient service.

Brief Description

In this system, the following features are included:

- **Rooms Details and Information:** All information about each room can easily be retrieved to guide staff on promoting and introducing the best options based on customer preferences.
- **Rooms Availability Management:** Provides real-time updates on room availability and blocks already unavailable rooms.
- **Guest Information Management:** Automatically stores guest information such as names, contact details, and booking history, facilitating check-in and check-out processes. It also tracks customers' preferences for repeated bookings.
- **Promos, Discounts, and Pricing Management:** Customizes promotional offers based on demand seasons and holidays.
- **Overall Reports Management:** Generates insights such as revenue trends and sales reports to aid in decision-making.

Objectives and Purpose

Purpose

The **AMRADIAMA:Hotel Management System** is a solution designed to streamline various hotel operations, minimizing manual errors and enhancing efficiency. By automating essential tasks such as reservations, promotions, and report generation, this system ensures a smoother experience for both staff and guests. Manual processes in hotels often lead to inefficiencies, errors, and delays. The purpose of this system is to mitigate these issues by providing a digital platform that centralizes operations. It serves as a reliable tool for small to medium-sized hotels, ensuring that data is accurate and easily accessible while enhancing user experience. It is an **Online Reservation System** with features that allows customers to book reservations online through the hotel's website or third-party platforms, integrated with secure booking transactions.

Objectives

The main objectives of the Hotel Management System include:

1. **Streamlining Hotel Operations:** Simplify the workflow of reservations, room management, and promotions through automation.
2. **Improving Data Accuracy:** Ensure that all records, such as reservations and guest details, are maintained accurately and up-to-date.
3. **Enhancing User Experience:** Provide a user-friendly interface that caters to both technical and non-technical users.
4. **Boosting Decision-Making:** Generate detailed reports for bookings and guest data to assist in strategic planning and management.

The Hotel Reservation Management System specifically aims to integrate seamlessly with the objectives listed earlier by:

- Simplifying the management of all rooms and accommodations for customers, ensuring efficiency and accuracy.
- Reducing overbooking and incorrect reservations, thereby minimizing errors and confusion while enhancing customer satisfaction.
- Offering real-time room availability updates, alleviating staff pressure and significantly improving operational workflow.

By addressing these objectives, the system empowers hotel staff to focus on delivering quality service while minimizing operational overhead.

Key Features

User Account Management

Managing user accounts is crucial for ensuring secure and efficient access to the system. This module includes:

- **Account Creation:** Allows users to register by entering their details such as name, contact information, and a secure password. This ensures a personalized experience for guests.
- **Login System:** Provides a secure authentication process where users enter their credentials to access the system. It prevents unauthorized access.
- **Account Deletion:** Permits users to permanently delete their accounts. This feature is particularly useful for maintaining data privacy and complying with user requests.

Source Code

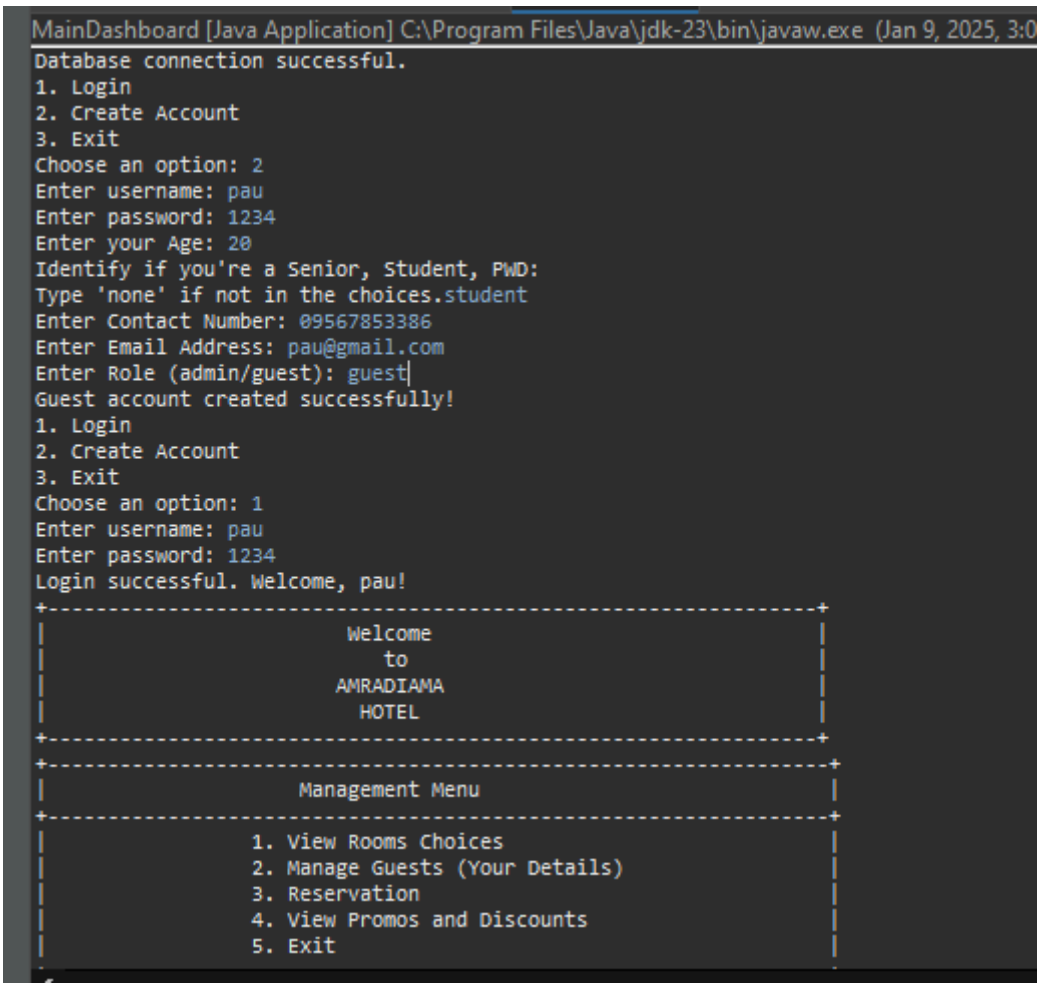
```java
package hotelmanagementSystem;
//creation of class admin
public class admins {
    private int id;
    private String username;
    private String password;

    // Getters and setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }
    public String getUsername() { return username; }
    public void setUsername(String username) { this.username = username; }
    public String getPassword() { return password; }
    public void setPassword(String password) { this.password = password; }
}
```

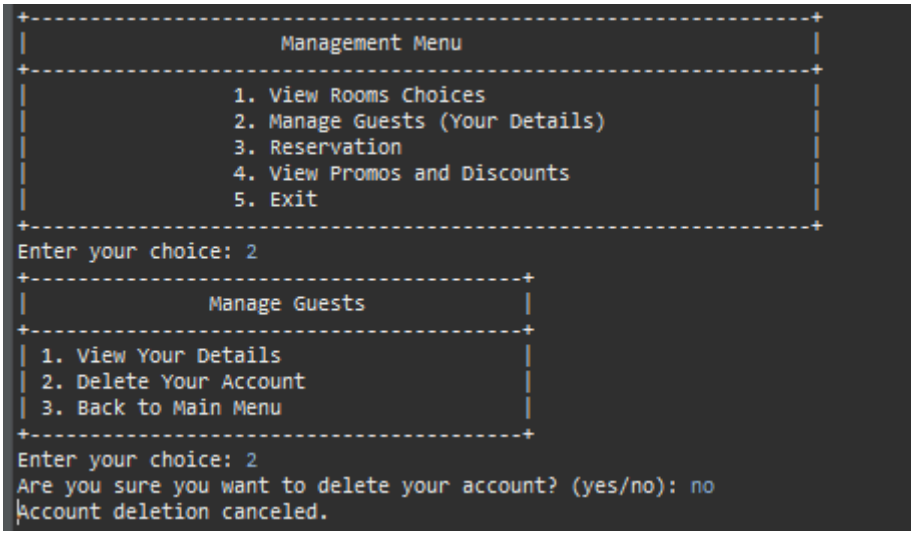
```java
public boolean login(String username, String password) {
    try {
        PreparedStatement stmt = conn.prepareStatement(
            // Checking if the inserted username and password are in the database
            "SELECT * FROM Guest WHERE Username = ? AND Password = ?"
        );
        stmt.setString(1, username);
        stmt.setString(2, password);
        ResultSet results = stmt.executeQuery();

        if (results.next()) {
            loggedInUser = username;
            System.out.println("Login successful. Welcome, " + username + "!");
            return true;
        } else {
            System.out.println("Login failed: Invalid username or password.");
            return false;
        }
    } catch (SQLException e) {
        System.out.println("Login failed: " + e.getMessage());
        return false;
    }
}
```

```java
    //initializing create account method for easy account creation
  private static void createGuestAccount(Scanner scanner) {
      // Get the username and password
      System.out.print("Enter username: ");
      String username = scanner.nextLine();

      System.out.print("Enter password: ");
      String password = scanner.nextLine();


      System.out.print("Enter your Age: ");
      int age = scanner.nextInt();
      scanner.nextLine();

   // Validate the customer type
      System.out.println("Identify if you're a Senior, Student, PWD: ");
      System.out.print("Type 'none' if not in the choices.");



      String customerType = scanner.nextLine().trim(); // Trim to remove extra spaces

   // Validate the CustomerType (case-insensitive)
   if (!customerType.equalsIgnoreCase("Senior") &&
       !customerType.equalsIgnoreCase("Student") &&
       !customerType.equalsIgnoreCase("PWD") &&
       !customerType.equalsIgnoreCase("none"))
   {
       System.out.println("Invalid CustomerType! Only 'Senior', 'Student', 'PWD', or 'none' is allowed.");
       return;
```

```java
  }



    System.out.print("Enter Contact Number: ");
    String contactNumber = scanner.nextLine();


    System.out.print("Enter Email Address: ");
    String emailAddress = scanner.nextLine();


    System.out.print("Enter Role (admin/guest): ");
    String role = scanner.nextLine().toLowerCase();

    // Validate the role
    if (!role.equals("admin") && !role.equals("guest")) {
        System.out.println("Invalid role! Only 'admin' or 'guest' is allowed.");
        return;
    }

    // Call the method to create the account
    loginSystem.createAccount(username, password, age, customerType, contactNumber, emailAddress, role);
}
```

Output





Room Reservation and Management

The reservation module is designed to simplify the process of booking and managing rooms. Its features include:

- **Book a Room:** Guests can select available rooms based on their preferences, such as room type, room number, and duration of stay. The system checks for availability and applies any active promotions.

Source Code

```java
package hotelmanagementSystem;
import java.sql.*;
// creating an interface class with its method
public interface Reservation {
    void bookARoom();
    void checkout();
}
//creating a class that implements the interface
class Booking implements Reservation {
    private static Connection conn;// connecting to the database
    private Scanner scanner;
    private LoginSystem loginSystem;
// constructor
    public Booking(Connection conn, Scanner scanner, LoginSystem loginSystem) {
        this.conn = conn;
        this.scanner = scanner;
        this.loginSystem = loginSystem;
    }

```

```java
@Override
public void bookARoom() {
    if (loginSystem.getLoggedInUser() == null) {// checking if there's a logged in user
        System.out.println("You must log in first.");
        return;
    }

    try {
        //checking of discount eligibility based on the logged in user customertype from the database
        System.out.println("Checking for discount eligibility...");
        System.out.println();

        String customerType = getCustomerType();
        if (customerType == null) return;

        double discountRate = getDiscountRate(customerType);
        displayDiscountEligibility(customerType, discountRate);

        double promoDiscountPercentage = processPromo();

        System.out.print("Enter the number of days you plan to stay: ");
        int numberOfDays = scanner.nextInt();
        scanner.nextLine();

        Map<String, Double> roomBookings = processRoomBookings(numberOfDays);

        double totalPrice = calculateTotalPrice(roomBookings, numberOfDays);
        double totalDiscount = totalPrice * promoDiscountPercentage / 100.0;
        double promoDiscount = totalPrice * discountRate;
        double finalPrice = totalPrice - promoDiscount - totalDiscount;
```

```java
        printBookingSummary(roomBookings, numberOfDays, totalPrice, totalDiscount, promoDiscount, finalPrice);

        if (confirmBooking()) {
            java.sql.Date checkInDate = getCheckInDate();
            if (checkInDate == null) return;

            if (saveBooking(roomBookings, totalDiscount, numberOfDays, finalPrice, checkInDate)) {
                updateRoomAvailability(roomBookings);
                System.out.println("Booking confirmed and room availability updated.\n");
                handlePayment(finalPrice);
            }
        } else {
            System.out.println("Booking cancelled.\n");
        }

    } catch (SQLException e) {
        System.out.println("Error during booking process: " + e.getMessage());
    }
}
```

```java
    private boolean isUserLoggedIn() {
        return loginSystem.getLoggedInUser() != null;
    }

    private String getCustomerType() throws SQLException {
        String query = "SELECT CustomerType FROM Guest WHERE Username = ?";
        try (PreparedStatement stmt = conn.prepareStatement(query)) {
            stmt.setString(1, loginSystem.getLoggedInUser());
            try (ResultSet rs = stmt.executeQuery()) {
                if (rs.next()) {
                    return rs.getString("CustomerType");
                } else {
                    System.out.println("Error: Customer details not found.");
                    return null;
                }
            }
        }
    }
```

```java
public double calculateTotalPrice(Map<String, Double> roomBookings, int numberOfDays) {
    double totalPrice = 0.0;

    for (double roomPrice : roomBookings.values()) {
        totalPrice += roomPrice; // Sum up room prices
    }

    return totalPrice * numberOfDays; // Multiply by the number of days
}


private void displayDiscountEligibility(String customerType, double discountRate) {
    if (discountRate > 0) {
        System.out.println("You are eligible for a discount as a " + customerType);
        System.out.println("Discount rate: " + (discountRate * 100) + "%.\n");
    } else {
        System.out.println("You are not eligible for any discounts.\n");
    }
}

private double processPromo() {
```

```java
private double processPromo() {
    System.out.print("Do you want to avail a promo? (yes/no): ");
    String response = scanner.nextLine().trim().toLowerCase();

    if (!response.equals("yes")) return 0.0;

    viewPromos();
    System.out.print("Enter the promo name you want to avail: ");
    String promoName = scanner.nextLine().trim();

    Promo promo = getPromoDetails(promoName);
    if (promo == null) {
        System.out.println("Invalid promo name or no such promo available.");
        return 0.0;
    }

    java.sql.Date checkInDate = getCheckInDate();
    if (checkInDate == null) return 0.0;

    if (isPromoValid(promo, checkInDate)) {
        System.out.println("Promo applied: " + promoName + " with " + promo.getDiscountPercentage() + "% discount.");
        return promo.getDiscountPercentage() * 100.0;
    } else {
        System.out.println("The promo is not valid for your selected check-in date.");
        return 0.0;
    }
}
```

```java
private boolean isPromoValid(Promo promo, java.sql.Date checkInDate) {
    return checkInDate.after(promo.getValidFrom()) && checkInDate.before(promo.getValidUntil());
}

private Map<String, Double> processRoomBookings(int numberOfDays) throws SQLException {
    Map<String, Double> roomBookings = new HashMap<>();

    while (true) {
        System.out.print("Enter room type or 'done' to finish: ");
        String roomType = scanner.nextLine();

        if (roomType.equalsIgnoreCase("done")) break;

        if (!isValidRoomType(roomType)) {
            System.out.println("Invalid room type. Please try again.");
            continue;
        }

        System.out.print("Enter the room number you want to book for " + roomType + ": ");
        int roomNumber = scanner.nextInt();
        scanner.nextLine();

        if (!isRoomAvailable(roomNumber, roomType)) {
            System.out.println("Room number " + roomNumber + " is not available.");
            continue;
        }

        double roomPrice = getRoomPrice(roomType);
        double calculatedPrice = roomPrice * numberOfDays;
        roomBookings.put(roomType + " (Room# " + roomNumber + ")", roomPrice);

        System.out.println("Room number " + roomNumber + " for " + roomType + " booked successfully.\n");
    }

    return roomBookings;
}
```

```java
private boolean confirmBooking() {
    System.out.print("Do you want to confirm the booking? (yes/no): ");
    String response = scanner.nextLine();
    return response.equalsIgnoreCase("yes");
}
```

Output

```
+-------------------------------------------------------------------+
Enter your choice: 3
+----------------------------------------+
|              Reservation Menu          |
+----------------------------------------+
| 1. View Reservation List               |
| 2. Book a Room                         |
| 3. Checkout                            |
| 4. Back to Main Menu                   |
+----------------------------------------+
Enter your choice: 2

Checking for discount eligibility...

You are eligible for a discount as a student
Discount rate: 15.0%.

Do you want to avail a promo? (yes/no): yes
+-------------------------------------------------------------------------------------------------------------------------------------------+
| Promo Name          | Discount (%) |                              Description                              |   Valid From   |  Valid Until   |
+-------------------------------------------------------------------------------------------------------------------------------------------+
| Holiday Special     | 20.00        | Enjoy a 20% discount when you book any room this San Pablo City Festival.     | 2025-01-12     |  2025-01-14    |
| New Year Special    | 15.00        | A 15% discount for first week of January bookings.                            | 2025-01-01     |  2025-01-04    |
+-------------------------------------------------------------------------------------------------------------------------------------------+
Enter the promo name you want to avail: holiday special
Enter check-in date (YYYY-MM-DD): 2025-01-13
Promo applied: holiday special with 0.2% discount.
Enter the number of days you plan to stay: 2
Enter room type or 'done' to finish: singleroom
+-----------------------------------------------+
|Room Number| Room Type       |  Availability  |
+-----------------------------------------------+
|101        |singleroom       |Available       |
|102        |singleroom       |Available       |
|103        |singleroom       |Available       |
```

```
|Room Number| Room Type       |  Availability  |
+-------------------------------------------+
|101        |singleroom       |Available       |
|102        |singleroom       |Available       |
|103        |singleroom       |Available       |
+-------------------------------------------+
Total rooms of type 'singleroom': 3
Enter the room number you want to book for singleroom: 101
Room number 101 for singleroom booked successfully.

Enter room type or 'done' to finish: done
+-------------------------------------------------+
| Description                   | Amount ($)      |
+-------------------------------------------------+
| singleroom (Room# 101)        | $100.00         |
+-------------------------------------------------+
| Days of staying               | 2               |
+-------------------------------------------------+
| Total Price                   | $200.00         |
| Promo Discount                | $40.00          |
| Total Discount                | $30.00          |
| Final Price to Pay            | $130.00         |
+-------------------------------------------------+
Do you want to confirm the booking? (yes/no): yes
Enter check-in date (YYYY-MM-DD): 2025-01-13
Booking confirmed and room availability updated.
```

- **View Reservations:** Users can view all current and past reservations, providing a clear overview of their booking history. This is particularly useful for staff when verifying guest details.

Source Code

```
}
//initializing view reservation method and getting the data on the database
private static void viewReservation() {
    // Correct query with appropriate column names
    String query = "SELECT booking_id, date, username, roomType, room_number, check_in_date FROM Bookings";

    try (Statement stmt = conn.createStatement(); ResultSet rs = stmt.executeQuery(query)) {
        // Print table header
        System.out.println("+------------------------------------------------------------------------------------------------------------+");
        System.out.println("| Booking ID |            Date            |    Username    |     Room Type      | Room Number | Check-in Date   |");
        System.out.println("+------------------------------------------------------------------------------------------------------------+");

        while (rs.next()) {
            // Extract data from ResultSet
            int bookingID = rs.getInt("booking_id");
            Timestamp date = rs.getTimestamp("date");
            String username = rs.getString("username");
            String roomType = rs.getString("roomType");
            int roomNumber = rs.getInt("room_number");
            Date checkInDate = rs.getDate("check_in_date");

          //displaying of the reservation data from the database with a design format
            System.out.printf("| %10d | %-27s | %-13s | %-18s | %11d | %-15s |%n",
                    bookingID,
                    date != null ? date.toString() : "N/A",
                    username,
                    roomType,
                    roomNumber,
                    checkInDate != null ? checkInDate.toString() : "N/A");
        }


        System.out.println("+------------------------------------------------------------------------------------------------------------+");
    } catch (SQLException e) {
        System.out.println("Error retrieving bookings: " + e.getMessage());
    }
```

Output

```
+----------------------------------------+
|              Reservation Menu          |
+----------------------------------------+
| 1. View Reservation List               |
| 2. Book a Room                         |
| 3. Checkout                            |
| 4. Back to Main Menu                   |
+----------------------------------------+
Enter your choice:
1
+-----------------------------------------------------------------------------------------------------------+
| Booking ID |            Date            |    Username    |     Room Type      | Room Number | Check-in Date   |
+-----------------------------------------------------------------------------------------------------------+
|          6 | 2024-12-25 00:00:00.0      | missdii        | singleroom         |         103 | 2025-01-01      |
+-----------------------------------------------------------------------------------------------------------+
```

- **Handling of Payment:** Automates the billing process by calculating the total amount based on room rates, additional charges, and discounts. It also updates the room status to "Available," ensuring accurate room inventory. In these features, guests can choose their payment method and process payment smoothly.

Source Code

```
private void handlePayment(double finalPrice) {
    System.out.println("+--------------------------------+");
    System.out.println("|        Payment Options         |");
    System.out.println("+--------------------------------+");
    System.out.println("| 1. Bank Card                   |");
    System.out.println("| 2. Online Wallet               |");
    System.out.println("| 3. Cash on Check-in Date       |");
    System.out.println("+--------------------------------+");
    System.out.print("Choose your payment method (1-3): ");

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

    switch (choice) {
        case 1:
            handleBankCardPayment(finalPrice);
            break;
        case 2:
            handleOnlineWalletPayment(finalPrice);
            break;
        case 3:
            System.out.println("You have chosen to pay with cash on the check-in date.");
            System.out.println("Please make sure to bring the exact amount: $" + finalPrice);
            break;
        default:
            System.out.println("Invalid choice. Please try again.");
            handlePayment(finalPrice);
            break;
    }
}
```

```java
private void handleBankCardPayment(double finalPrice) {
    System.out.println("Please enter your bank card details:");
    System.out.print("Card Number: ");
    String cardNumber = scanner.nextLine();
    System.out.print("Expiration Date (MM/YY): ");
    String expirationDate = scanner.nextLine();
    System.out.print("CVV: ");
    String cvv = scanner.nextLine();

    // Simulate payment processing
    System.out.println("Processing payment...");
    System.out.println("Payment of $" + finalPrice + " completed successfully using Bank Card.");
}
```

```java
private void handleOnlineWalletPayment(double finalPrice) {
    System.out.println("Please choose your online wallet:");
    System.out.println("1. Gcash");
    System.out.println("2. Maya");
    System.out.println("3. Paypal");
    System.out.print("Enter your choice: ");

    int walletChoice = scanner.nextInt();
    scanner.nextLine(); // Consume the newline character

    System.out.print("Enter your wallet account number: ");
    String walletID = scanner.nextLine();

    // Simulate payment processing
    System.out.println("Processing payment...");
    System.out.println("Payment of $" + finalPrice + " completed successfully using Online Wallet.");
}
```

Output

```
+--------------------------------+
|        Payment Options         |
+--------------------------------+
| 1. Bank Card                   |
| 2. Online Wallet               |
| 3. Cash on Check-in Date       |
+--------------------------------+
Choose your payment method (1-3): 2
Please choose your online wallet:
1. Gcash
2. Maya
3. Paypal
Enter your choice: 2
Enter your wallet account number: 09567853386
Processing payment...
Payment of $130.0 completed successfully using Online Wallet.
+----------------------------------------+
```

- **Check Out:** A check out process wherein upon entering the room number you want to check out, the system will automatically check the availability of the room validating if the room was booked or not. It also allows the user to think twice on their decision because of the confirmation asked by the system.

Source Code

```java
@Override
public void checkout() {
    System.out.print("Enter Room Number to check out: ");
    int roomNumber = scanner.nextInt();
    scanner.nextLine(); // consume the newline character

    // Prompt for confirmation before proceeding with the checkout
    System.out.print("Are you sure you want to check out? (yes/no): ");
    String confirmation = scanner.nextLine().trim().toLowerCase();

    if (!confirmation.equals("yes")) {
        System.out.println("Checkout cancelled.");
        return; // Exit the method if the user doesn't confirm
    }

    // Checking if the room number entered by the user exists in the database
    String checkQuery = "SELECT isAvailable FROM Rooms WHERE roomNumber = ?";

    try (PreparedStatement checkStmt = conn.prepareStatement(checkQuery)) {
        checkStmt.setInt(1, roomNumber);
        try (ResultSet rs = checkStmt.executeQuery()) {
            // If the room number is found and set to "Not Available", it's confirmed as booked
            if (rs.next() && "Not Available".equalsIgnoreCase(rs.getString("isAvailable"))) {
                // Get the current date for checkout
                Date checkOutDate = new Date(System.currentTimeMillis());

                // Update the room availability to "Available" and set the checkout date in the bookings table
                String updateRoomQuery = "UPDATE Rooms SET isAvailable = ? WHERE roomNumber = ?";
                try (PreparedStatement updateRoomStmt = conn.prepareStatement(updateRoomQuery)) {
                    updateRoomStmt.setString(1, "Available");
                    updateRoomStmt.setInt(2, roomNumber);
                    updateRoomStmt.executeUpdate();
                }
```

```java
                String updateBookingQuery = "UPDATE Bookings SET check_out_date = ? WHERE room_number = ?";
                try (PreparedStatement updateBookingStmt = conn.prepareStatement(updateBookingQuery)) {
                    updateBookingStmt.setDate(1, checkOutDate);
                    updateBookingStmt.setInt(2, roomNumber);
                    updateBookingStmt.executeUpdate();
                }

                System.out.println("Room " + roomNumber + " has been successfully checked out.");
            } else {
                System.out.println("Room " + roomNumber + " is not currently booked.");
            }
        }
    } catch (SQLException e) {
        System.out.println("Error during checkout: " + e.getMessage());
    }
}
```

Output

```
+----------------------------------------+
|              Reservation Menu          |
+----------------------------------------+
| 1. View Reservation List               |
| 2. Book a Room                         |
| 3. Checkout                            |
| 4. Back to Main Menu                   |
+----------------------------------------+
Enter your choice: 3

Enter Room Number to check out: 101
Are you sure you want to check out? (yes/no): yes
Room 101 is not currently booked.
+----------------------------------------+
```

Additionally, the system includes:

- **Room Details and Information:** Allows staff to retrieve detailed information about each room to guide guests in selecting the best option based on their preferences.

Source Code

```
//initializing viewrooms method and getting needed data form the database
    private static void viewRooms() {
        String query = "SELECT roomNumber, roomType, price, isAvailable FROM Rooms";
//displaying the get data from the database with a design format
        try (Statement stmt = conn.createStatement(); ResultSet rs = stmt.executeQuery(query)) {
            System.out.println("+---------------------------------------------------------+");
            System.out.println("| Room Number | Room Type    | Price   | Availability      |");
            System.out.println("+---------------------------------------------------------+");

            while (rs.next()) {
                int roomNumber = rs.getInt("roomNumber");
                String roomType = rs.getString("roomType");


                double price = rs.getDouble("price");


                String availability = rs.getString("isAvailable");


                System.out.printf("| %11d | %-12s | %7.2f | %-15s |%n", roomNumber, roomType, price, availability);
            }

            System.out.println("+---------------------------------------------------------+");
        } catch (SQLException e) {
            System.out.println("Error retrieving rooms: " + e.getMessage());
        }
    }
}
```

Output

```
MainDashboard [Java Application] C:\Program Files\Java\jdk-23\bin\javaw.exe (Dec 27, 20
+-------------------------------------------------------------------+
|                        Management Menu                            |
+-------------------------------------------------------------------+
|                  1. View Rooms Choices                            |
|                  2. Manage Guests (Your Details)                  |
|                  3. Reservation                                   |
|                  4. View Promos and Discounts                     |
|                  5. Exit                                          |
+-------------------------------------------------------------------+
Enter your choice: 1
+---------------------------------------------------------+
| Room Number | Room Type    | Price   | Availability      |
+---------------------------------------------------------+
|         101 | SingleRoom   |  100.00 | available         |
|         102 | SingleRoom   |  100.00 | available         |
|         103 | SingleRoom   |  100.00 | Not Available     |
|         104 | CoupleRoom   |  150.00 | available         |
|         105 | CoupleRoom   |  150.00 | available         |
|         106 | CoupleRoom   |  150.00 | available         |
|         107 | SuiteRoom    |  200.00 | available         |
|         108 | SuiteRoom    |  200.00 | available         |
|         109 | SuiteRoom    |  200.00 | available         |
|         110 | DeluxeRoom   |  250.00 | available         |
|         111 | DeluxeRoom   |  250.00 | available         |
|         112 | DeluxeRoom   |  250.00 | available         |
|         113 | StudioRoom   |  300.00 | available         |
|         114 | StudioRoom   |  300.00 | available         |
|         115 | StudioRoom   |  300.00 | available         |
|         116 | FamilyRoom   |  350.00 | available         |
|         117 | FamilyRoom   |  350.00 | available         |
|         118 | FamilyRoom   |  350.00 | available         |
|         119 | VIPRoom      |  400.00 | available         |
|         120 | VIPRoom      |  400.00 | available         |
|         121 | FamilyRoom   |  400.00 | available         |
|         122 | EconomyRoom  |  450.00 | available         |
<
```

- **Rooms Availability Management:** Provides real-time updates on room availability and automatically marks unavailable rooms.

Source Code

```java
private boolean isRoomAvailable(int roomNumber, String roomType) {
    String query = "SELECT isAvailable FROM Rooms WHERE roomNumber = ? AND roomType = ?";
    try (PreparedStatement stmt = conn.prepareStatement(query)) {
        stmt.setInt(1, roomNumber);
        stmt.setString(2, roomType);

        try (ResultSet rs = stmt.executeQuery()) {
            if (rs.next()) {
                return "available".equalsIgnoreCase(rs.getString("isAvailable"));
            }
            System.out.println("Room number " + roomNumber + " for " + roomType + " does not exist.");
            return false;
        }
    } catch (SQLException e) {
        System.out.println("Error checking room availability: " + e.getMessage());
    }
    return false;
}
```

```java
//creation of method for updating of room availability
    private void updateRoomAvailability(Map<String, Double> roomBookings) {
        String updateQuery = "UPDATE Rooms SET isAvailable = ? WHERE roomNumber = ? AND roomType = ?";
        try (PreparedStatement stmt = conn.prepareStatement(updateQuery)) {
            for (String roomDescription : roomBookings.keySet()) {
                int roomNumber = extractRoomNumber(roomDescription);
                String roomType = extractRoomType(roomDescription);


                stmt.setString(1, "Not Available");
                stmt.setInt(2, roomNumber);
                stmt.setString(3, roomType);
                stmt.executeUpdate();

                //confirmation display of room availability setting
                System.out.println("Updated Room# " + roomNumber + " (" + roomType + ") to Not Available.");
            }
            System.out.println("Room availability updated successfully.");
        } catch (SQLException e) {
            System.out.println("Error updating room availability: " + e.getMessage());
        }
    }
```

Output

```
MainDashboard [Java Application] C:\Program Files\Java\jdk-23\bin\javaw.exe (Dec 27, 20
+-------------------------------------------------------------------+
|                        Management Menu                            |
+-------------------------------------------------------------------+
|                  1. View Rooms Choices                            |
|                  2. Manage Guests (Your Details)                  |
|                  3. Reservation                                   |
|                  4. View Promos and Discounts                     |
|                  5. Exit                                          |
+-------------------------------------------------------------------+
Enter your choice: 1
+-------------------------------------------------------------+
| Room Number | Room Type     | Price    | Availability       |
+-------------------------------------------------------------+
|         101 | SingleRoom    |   100.00 | available          |
|         102 | SingleRoom    |   100.00 | available          |
|         103 | SingleRoom    |   100.00 | Not Available      |
|         104 | CoupleRoom    |   150.00 | available          |
|         105 | CoupleRoom    |   150.00 | available          |
|         106 | CoupleRoom    |   150.00 | available          |
|         107 | SuiteRoom     |   200.00 | available          |
|         108 | SuiteRoom     |   200.00 | available          |
|         109 | SuiteRoom     |   200.00 | available          |
|         110 | DeluxeRoom    |   250.00 | available          |
|         111 | DeluxeRoom    |   250.00 | available          |
|         112 | DeluxeRoom    |   250.00 | available          |
|         113 | StudioRoom    |   300.00 | available          |
|         114 | StudioRoom    |   300.00 | available          |
|         115 | StudioRoom    |   300.00 | available          |
|         116 | FamilyRoom    |   350.00 | available          |
|         117 | FamilyRoom    |   350.00 | available          |
|         118 | FamilyRoom    |   350.00 | available          |
|         119 | VIPRoom       |   400.00 | available          |
|         120 | VIPRoom       |   400.00 | available          |
|         121 | FamilyRoom    |   400.00 | available          |
|         122 | EconomyRoom   |   450.00 | available          |
<
```

Guest Information Management

This module ensures efficient organization of guest data:

- Automatically stores guest details, such as name, contact information, and booking history.
- Tracks customer preferences for personalized service during repeat bookings.

Source Code

```
//creation of account
public void createAccount(String username, String password, int age, String customerType,
        String Contactnumber    , String emailAddress, String role) {
    try {
        //checking if the username inputted was already on the database, to display an error
        String checkUserQuery = "SELECT * FROM Guest WHERE Username = ?";
        PreparedStatement checkUserStmt = conn.prepareStatement(checkUserQuery);
        checkUserStmt.setString(1, username);
        ResultSet resultSet = checkUserStmt.executeQuery();

        if (resultSet.next()) {
            System.out.println("Error: Username already exists. Please choose a different username.");
            return;
        }

        //saving the the data inputed to guest table on the database.
        String query = "INSERT INTO Guest (Username, Password, Age, CustomerType, Contactnumber, EmailAddress, role) VALUES (?, ?, ?, ?, ?, ?, ?)";
        try (PreparedStatement stmt = conn.prepareStatement(query)) {
            stmt.setString(1, username);
            stmt.setString(2, password);
            stmt.setInt(3, age);
            stmt.setString(4, customerType);
            stmt.setString(5, Contactnumber );
            stmt.setString(6, emailAddress);
            stmt.setString(7, role);

            stmt.executeUpdate();
            System.out.println(role.substring(0, 1).toUpperCase() + role.substring(1) + " account created successfully!");
        }
    } catch (SQLException e) {
        System.out.println("Error creating account: " + e.getMessage());
    }
}
```

Output

```
+-----------------------------------------+
|              Manage Guests              |
+-----------------------------------------+
| 1. View Your Details                    |
| 2. Delete Your Account                  |
| 3. Back to Main Menu                    |
+-----------------------------------------+
Enter your choice: 1
+-------------------------------------------------------------------------------+
| Username   | Age   | Customer Type |   Contact   |            Email           |
+-------------------------------------------------------------------------------+
| missdii    | 26    | student       | 09567853386 | missdii@gmail.com          |
+-------------------------------------------------------------------------------+
```

Promotions Management

Promotional offers are a powerful tool for attracting guests. This module enables hotel staff to manage promotions efficiently:

- **Add Promotion:** Staff can create new promotions by specifying details such as discount percentage, validity period, and applicable room types.
- **Update Promotion:** Allows staff to modify existing promotions to reflect changes in offers or conditions.

- **Delete Promotion:** Removes expired or irrelevant promotions from the system, keeping the list up-to-date.
- **View Promotions:** Displays all active promotions, making it easy for guests to avail of special offers.

Promos, Discounts, and Pricing Management

This feature allows:

- Customization of promos, discounts, and vouchers based on demand seasons and holidays.
- Automatic display of promotional offers on screens for online users.

Source Code

```
//initializing managepromos method and displaying its menu
    private static void managePromos(Scanner scanner) {
        while (true) {
            System.out.println("+-----------------------------------------+");
            System.out.println("|            Manage Promos                |");
            System.out.println("+-----------------------------------------+");
            System.out.println("| 1. Add Promo                            |");
            System.out.println("| 2. Update Promo                         |");
            System.out.println("| 3. Delete Promo                         |");
            System.out.println("| 4. Back to Promo Menu                   |");
            System.out.println("+-----------------------------------------+");
            System.out.print("Enter your choice: ");
            int adminChoice = scanner.nextInt();
            scanner.nextLine();

            switch (adminChoice) {
                case 1:
                    addPromo(scanner);// calling the addpromo method
                    break;
                case 2:
                    updatePromo(scanner); // calling the update promo method
                    break;
                case 3:
                    deletePromo(scanner); // calling the deletepromo method
                    break;
                case 4:
                    System.out.println("Returning to Promo Menu...");
                    return;
                default:
                    System.out.println("Invalid choice. Please try again.");
            }
        }
    }
    //initializing the addpromo method that create new promo
```

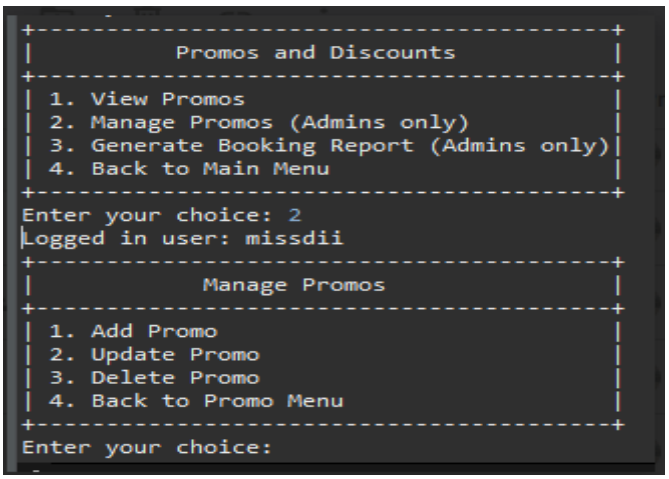
```
//initializing the addpromo method that create new promo
private static void addPromo(Scanner scanner) {
    System.out.print("Enter promo name: ");
    String promoName = scanner.nextLine();
    System.out.print("Enter discount percentage: ");
    double discount = scanner.nextDouble();
    scanner.nextLine(); // Consume newline
    System.out.print("Enter promo description: ");
    String description = scanner.nextLine();
    System.out.print("Enter validity date (YYYY-MM-DD): ");
    String validityDate = scanner.nextLine();
    //saving the new promo on database promos table
    String query = "INSERT INTO Promos (promo_name, discount_percentage, description, validity_date) VALUES (?, ?, ?, ?)";

    try (PreparedStatement stmt = conn.prepareStatement(query)) {
        stmt.setString(1, promoName);
        stmt.setDouble(2, discount);
        stmt.setString(3, description);
        stmt.setDate(4, Date.valueOf(validityDate));
        stmt.executeUpdate();
        System.out.println("Promo added successfully!");
    } catch (SQLException e) {
        System.out.println("Error adding promo: " + e.getMessage());
    }
}
```

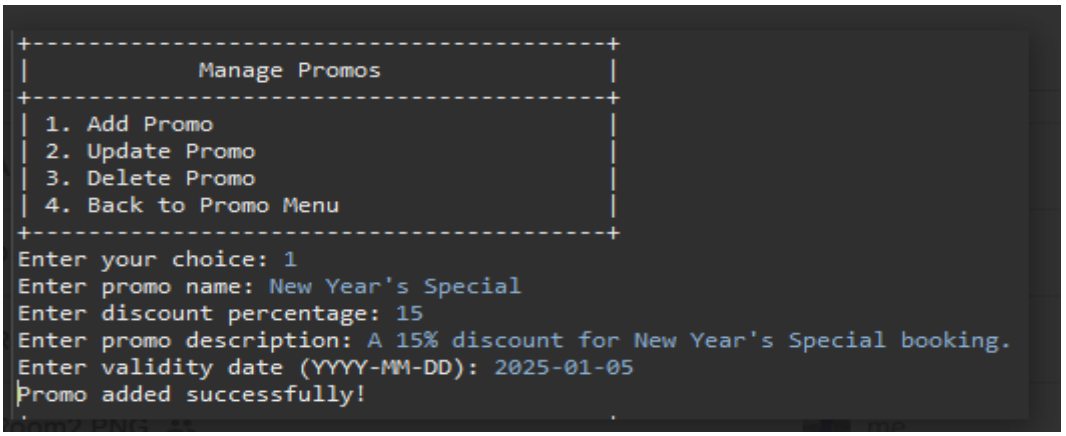
```java
//initializing update promo method to edit an existing promo
private static void updatePromo(Scanner scanner) {
    System.out.print("Enter promo ID to update: ");
    int promoId = scanner.nextInt();
    scanner.nextLine(); // Consume newline
    System.out.print("Enter new promo name: ");
    String promoName = scanner.nextLine();
    System.out.print("Enter new discount percentage: ");
    double discount = scanner.nextDouble();
    scanner.nextLine(); // Consume newline
    System.out.print("Enter new promo description: ");
    String description = scanner.nextLine();
    System.out.print("Enter new validity date (YYYY-MM-DD): ");
    String validityDate = scanner.nextLine();
    //updating the promo on database
    String query = "UPDATE Promos SET promo_name = ?, discount_percentage = ?, description = ?, validity_date = ? WHERE id = ?";

    try (PreparedStatement stmt = conn.prepareStatement(query)) {
        stmt.setString(1, promoName);
        stmt.setDouble(2, discount);
        stmt.setString(3, description);
        stmt.setDate(4, Date.valueOf(validityDate));
        stmt.setInt(5, promoId);
        stmt.executeUpdate();
        System.out.println("Promo updated successfully!");
    } catch (SQLException e) {
        System.out.println("Error updating promo: " + e.getMessage());
    }
}
//initializing delete promo method and remove a promo on the database
```

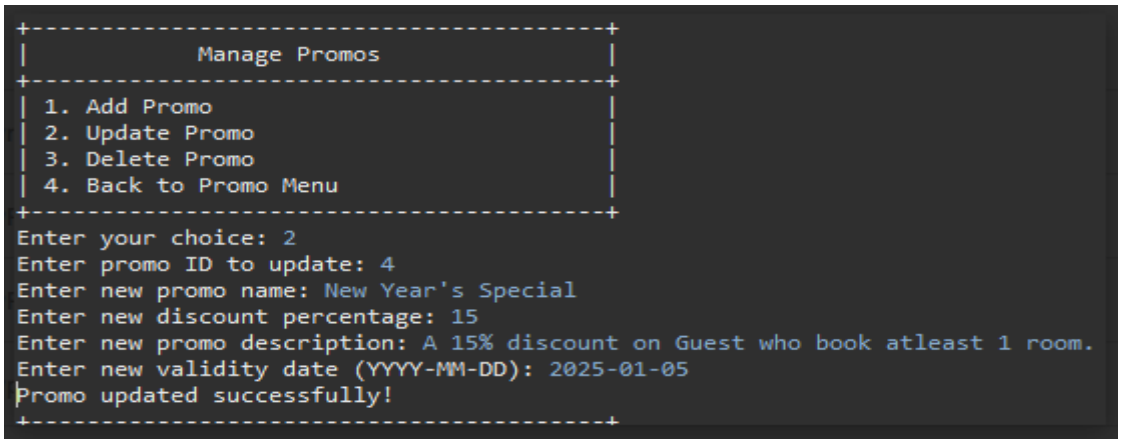
```java
//initializing delete promo method and remove a promo on the database
    private static void deletePromo(Scanner scanner) {
        System.out.print("Enter promo ID to delete: ");
        int promoId = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        String query = "DELETE FROM Promos WHERE id = ?";

        try (PreparedStatement stmt = conn.prepareStatement(query)) {
            stmt.setInt(1, promoId);
            int rowsAffected = stmt.executeUpdate();
            if (rowsAffected > 0) {
                System.out.println("Promo deleted successfully!");
            } else {
                System.out.println("Promo not found.");
            }
        } catch (SQLException e) {
            System.out.println("Error deleting promo: " + e.getMessage());
        }
    }
```

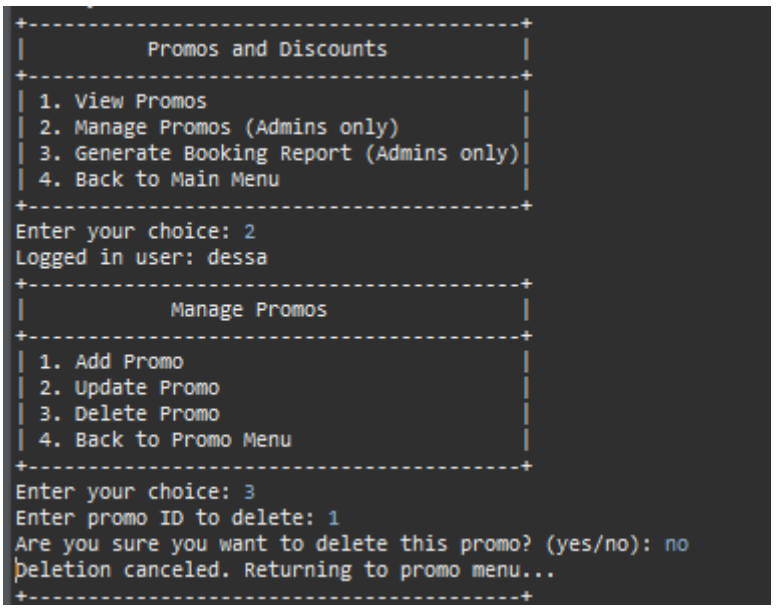
Output

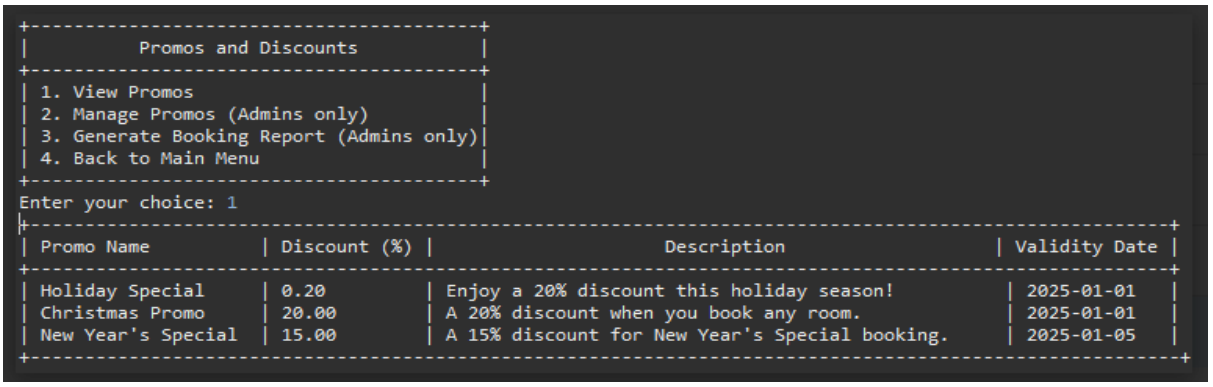
```
+-----------------------------------------+
|          Promos and Discounts           |
+-----------------------------------------+
| 1. View Promos                          |
| 2. Manage Promos (Admins only)          |
| 3. Generate Booking Report (Admins only)|
| 4. Back to Main Menu                    |
+-----------------------------------------+
Enter your choice: 2
Logged in user: pau
Access denied. Only admins can manage promos.
+-------------------------------------------------------------------+
|                       Management Menu                             |
+-------------------------------------------------------------------+
|                  1. View Rooms Choices                            |
|                  2. Manage Guests (Your Details)                  |
|                  3. Reservation                                   |
|                  4. View Promos and Discounts                     |
|                  5. Exit                                          |
+-------------------------------------------------------------------+
Enter your choice:
```


+--+
| Promos and Discounts |
+--+
| 1. View Promos |
| 2. Manage Promos (Admins only) |
| 3. Generate Booking Report (Admins only)|
| 4. Back to Main Menu |
+--+
Enter your choice: 2
Logged in user: missdii
+--+
| Manage Promos |
+--+
| 1. Add Promo |
| 2. Update Promo |
| 3. Delete Promo |
| 4. Back to Promo Menu |
+--+
Enter your choice:


+--+
| Manage Promos |
+--+
| 1. Add Promo |
| 2. Update Promo |
| 3. Delete Promo |
| 4. Back to Promo Menu |
+--+
Enter your choice: 1
Enter promo name: New Year's Special
Enter discount percentage: 15
Enter promo description: A 15% discount for New Year's Special booking.
Enter validity date (YYYY-MM-DD): 2025-01-05
Promo added successfully!


+--+
| Manage Promos |
+--+
| 1. Add Promo |
| 2. Update Promo |
| 3. Delete Promo |
| 4. Back to Promo Menu |
+--+
Enter your choice: 2
Enter promo ID to update: 4
Enter new promo name: New Year's Special
Enter new discount percentage: 15
Enter new promo description: A 15% discount on Guest who book atleast 1 room.
Enter new validity date (YYYY-MM-DD): 2025-01-05
Promo updated successfully!
+--+





Reporting Managements

The reporting module provides critical insights into hotel operations:

- **Generate Booking Report:** Summarizes all bookings within a specified time frame, providing a detailed view of occupancy rates and revenue.
- **Generate Guest Booking Report:** Focuses on individual guests, detailing their booking history and preferences. This helps in personalizing guest experiences.
- **Overall Reports Management:** Generates insights such as revenue trends and sales reports to aid in decision-making.

Source Code

```java
//initializing generatebookingreport method
private static void generateBookingReport() {
    // SQL query to fetch the summarized booking report on database tables
    String query = "SELECT " +
                   "COUNT(b.booking_id) AS total_booked_rooms, " +
                   "SUM(r.price) AS total_price, " +
                   "GROUP_CONCAT(b.room_number ORDER BY b.room_number) AS booked_room_numbers, " +
                   "GROUP_CONCAT(r.roomType ORDER BY b.room_number) AS booked_room_types, " +
                   "GROUP_CONCAT(g.Username ORDER BY b.room_number) AS guest_names " +
                   "FROM Bookings b " +
                   "JOIN Guest g ON g.Username = b.username " +
                   "JOIN Rooms r ON b.room_number = r.roomNumber";

    try (Statement stmt = conn.createStatement(); ResultSet rs = stmt.executeQuery(query)) {
        if (rs.next()) {
            // Get the summarized data
            int totalBookedRooms = rs.getInt("total_booked_rooms");
            double totalPrice = rs.getDouble("total_price");
            String bookedRoomNumbers = rs.getString("booked_room_numbers");
            String bookedRoomTypes = rs.getString("booked_room_types");
            String guestNames = rs.getString("guest_names");

            // displaying the Output summary report
            System.out.println("+-------------------------------------------------------------------------+");
            System.out.println("| Booking Summary Report                                                  |");
            System.out.println("+-------------------------------------------------------------------------+");
            System.out.printf("| Total Booked Rooms       | %-36d |\n", totalBookedRooms);
            System.out.printf("| Total Price              | $%-35.2f |\n", totalPrice);
            System.out.println("+-------------------------------------------------------------------------+");
            System.out.printf("| Booked Room Numbers      | %-36s |\n", bookedRoomNumbers);
            System.out.printf("| Booked Room Types        | %-36s |\n", bookedRoomTypes);
            System.out.printf("| Guest Names              | %-36s |\n", guestNames);
            System.out.println("+-------------------------------------------------------------------------+");
```

Output

```
+-----------------------------------------+
|          Promos and Discounts           |
+-----------------------------------------+
| 1. View Promos                          |
| 2. Manage Promos (Admins only)          |
| 3. Generate Booking Report (Admins only)|
| 4. Back to Main Menu                    |
+-----------------------------------------+
Enter your choice: 3
Logged in user: dessa
+-------------------------------------------------------------------------+
| Booking Summary Report                                                  |
+-------------------------------------------------------------------------+
| Total Booked Rooms       | 1                                            |
| Total Price              | $100.00                                      |
+-------------------------------------------------------------------------+
| Booked Room Numbers      | 101                                          |
| Booked Room Types        | SingleRoom                                   |
| Guest Names              | pau                                          |
+-------------------------------------------------------------------------+
```

OOP Concepts to be Implemented

This system implements key Object-Oriented Programming (OOP) principles:

- **Encapsulation:** Ensures sensitive guest information, Guest payment details and other data that is needed to hide is securely stored and accessed.
- **Polymorphism:** Supports overloading and overriding of interface methods and other methods with the help of getter and setter methods.
- **Abstraction:** Simplifies complex processes like reservation adjustments, discounts, and promotions into user-friendly interfaces.

Database or Storage Method

The system utilizes a relational database management system (RDBMS), specifically MySQL. This choice ensures:

- **Structured Storage:** Allows for efficient organization and retrieval of data.
- **Scalability:** Accommodates growing data needs as the hotel expands.
- **Real-Time Updates:** Ensures data integrity and synchronization across all modules.

Conclusion

The Hotel Management System is a comprehensive tool designed to address the unique challenges faced by hotels. Its features ensure efficient handling of core operations, from reservations to promotions, while minimizing errors and saving time. It also has the capabilities to offer smooth and secured payment transactions with friendly discounts and promos. It also possesses security on most of sensitive data in a management system that will provide high appreciation on the users and owners.

With the implementation of this system, hotels can offer a superior guest experience by automating tedious tasks and focusing on quality service.The reporting capabilities provide valuable insights, enabling management to make data-driven decisions and improve operational strategies.

Future enhancements, such as the integration of a graphical user interface and mobile application support, will further elevate the system’s functionality. For example, implementing a graphical user interface would allow staff to intuitively navigate the system, reducing training time and errors. A mobile application would

enable guests to make bookings, check room availability, and receive personalized offers directly on their smartphones, enhancing convenience and customer satisfaction. By continually evolving, the Hotel Management System aims to remain a reliable and indispensable tool for the hospitality industry.

In conclusion, this system represents a significant step forward in hotel management, providing a scalable and robust solution that adapts to the dynamic needs of the industry.