Programming Patterns Project

Our project will be a Travel Agency System (TAS) meant to make booking reservations easier and more efficient. This system will help manage bookings for flights and hotels, and customer service, all in one place. The system will provide an easy-to-use platform for customers to send a request for booking and to manage their flight tickets and hotel reservations if they have one, with the help of the employees of the agency. It will also store customer profiles, flight details, and payment information in different locations. The customers will also be able to leave a review after booking a flight to let the agency know about their experience as a customer. The booking system allows customers to book and cancel their booking, but they cannot modify their flight without going through the agency. They need to send a request and the employees are going to be in charge of modifying their flights with their updated preferences depending on the availability. We also are going to keep track of customer profiles, employee profiles, payment history, and tickets bought and canceled on a database. For the payment processing, the client is going to have 2 options to handle their payment, either by credit card or with their points accumulated, since they gain points every time they purchase a ticket, depending on the cost of their ticket. The employee can also buy their own flight tickets and hotel reservations, and they have the same payment processing but they are also eligible for a discount on it. The discount percentage depends on their role in the company (regular employee or manager). When canceling a flight, the client gets a refund either on their credit card or they get their points back if they paid it that way.

Expected Results

1. **Search for Flights**:
    - **Expected Output**: The system returns a list of available flights matching the search criteria. Each flight displays details such as:
        - Flight number
        - Airline name
        - Departure and arrival times
        - Price per seat
    - **Additional Output**: If no flights match the search criteria, the system displays a message like: "No flights available for the selected criteria."
2. **Book a Flight**:
    - **Expected Output**: The system confirms the booking, assigns the ticket to the passenger by adding it to the list of booked tickets, including:
        - Booking number

        - Flight details (flight number, departure, and arrival times)
        - Passenger details
        - Total cost
3. **Book a Room**:
    - **Expected Output**: The system confirms the booking, and assigns the room to the client by adding it to the list of booked rooms, including:
        - room number
        - Booking details (number of nights, price, if food is included, and arrival times and departure times)
        - client details
        - Total cost
4. **Cancel a Flight Booking**:
    - **Expected Output**: The system cancels the booking and confirms the cancellation, updating the list of booked tickets and adding it to canceled tickets. A message like "Booking successfully canceled" should be displayed.
    - **Additional Output**: If the booking reference is invalid, the system displays: "Booking not found. Please check your booking number."
5. **Cancel a Hotel Booking**:
        - **Expected Output**: The system cancels the booking and confirms the cancellation, updating the list of booked rooms and adding it to the list of available rooms. A message like "Booking successfully canceled" should be displayed.
6. **View Booking Details**:
    - **Expected Output**: The system displays all relevant booking details, including passenger information, flight details, and the total price. If the booking is not found, the system should show an error message.
7. **Payment and Invoice Generation**:
    - **Expected Output**: The system generates an invoice after the payment is processed, displaying the total amount paid, payment method, and ticket details.
8. **View Flight Information**:
    - **Expected Output**: The system displays a list of upcoming flights.
9. **View Rooms Availabilities**:
        - **Expected Output**: The system displays a list of booked hotel rooms and available hotel rooms.
10. **Error Handling**:
    - **Expected Output**: For each invalid input, the system provides specific feedback, like "Please enter a valid email" or “Ticket ID not found."

Design Paradigm

Flight

1. **Search for Flights**:

    - **Action**: Users (customers or agents) can search for available flights based on departure city, destination city, date, and class of service (e.g., economy, business).
2. **Book a Flight**:
    - **Action**: Agents can select a flight from the search results and proceed to book a seat by providing passenger details (name, age, passport number, phone number and email).
3. **Cancel a Booking**:
    - **Action**: Agents can cancel an existing booking by the booking reference.
4. **View Booking Details**:
    - **Action**: Users (customers and agents) can view details of an existing booking by entering their booking number.
5. **Payment and Invoice Generation**:
    - **Action**: Employees can proceed to pay for the flight of the client using their credit card number. Payment can be simulated through a simple process.
6. **View Flight Information**:
    - **Action**: Users or agents can view general flight information, including upcoming departures, flight status, and available seats without making a booking.
7. **Error Handling**:
    - **Action**: If users provide invalid inputs (e.g., wrong dates, incomplete passenger details, invalid booking number), the system should display clear error messages.

<u>Hotel</u>

8. **Search for Available Rooms**:
    - **Action**: Users (customers or agents) can search for available rooms based on availability in the hotel.
9. **Reserve a Room**:
    - **Action**: Agents can select a room from the search results and proceed to book a room by providing the client's details (name, age, phone number and email).
10. **Cancel a Booking**:
    - **Action**: Agents can cancel an existing booking by the room number and client name.
11. **View Booking Details**:
    - **Action**: Users (customers and agents) can view details of an existing booking.
12. **Payment and Invoice Generation**:
    - **Action**: Employees can proceed to pay the flight of the client using their credit card number. Payment can be simulated through a simple process.
13. **View Hotel Information**:
    - **Action**: Users or agents can view general hotel information without making a booking.
14. **Error Handling**:
    - **Action**: If users provide invalid inputs (e.g., wrong dates, incomplete client details, invalid user id), the system should display clear error messages.

Overall Behavior:

- The system should be responsive and provide clear, concise messages for both successful actions and errors.
- Admin-specific features should be password-protected or restricted to certain users.
- The system should ensure data integrity, for example, by preventing overbooking, double bookings, or incorrect flight details.

* Git Repository: Initialize a Maven project with valid `.gitignore`, and a `README.md` file for a project description. Create a `doc` folder which contains diagrams and the Deliverable 1 PDF.

Git Link: https://github.com/shahzaib786ahmed/programming-patterns.git