Project Proposal: Flight Booking Management System (CLI)

1. Project Title

Flight Booking Management System (FBMS)

2. Objective

To develop a command-line interface (CLI) application for managing flight bookings. The system allows users to add flights and passengers, book and cancel tickets, view and search for flights, and search for passenger details.

3. Scope

The system will support two main user roles: Admin and Passengers. Admins can manage flights, while passengers can book and manage their flights.

4. Features

1. Flight Management

- **Add Flight**: Add new flights with details such as flight number, origin, destination, schedule, and seat capacity.
- **Remove Flight**: Remove flights based on the flight number.
- **Update Flight Details**: Modify flight schedules and seat availability.
- **View Flights**: Display all available flights.

2. Passenger Management

- **Add Passenger**: Register passengers with personal details like name, age, and passport number.
- **Remove Passenger**: Remove passenger records.
- **Update Passenger Details**: Update passenger contact information.
- **View Passenger Details**: View detailed information for each passenger.

3. Booking Management

- **Book Ticket**: Book a ticket for a specific flight.
- **Cancel Ticket**: Cancel an existing booking.

4. Search Functionality

- **Search Flights**: Search for flights based on origin, destination, and whether they are domestic or international.
- **Search Passengers**: Search for passengers and view details such as name, age, passport number, flights taken, destinations, number of tickets bought, and flight type (domestic or international).

5. Report Generation

- **Booking Reports**: Generate reports showing booking details for each flight.
- **Flight Performance Reports**: Provide insights on seat occupancy and booking trends.

5. Technologies Used

- **Programming Language**: Java
- **Development Environment**: IntelliJ IDEA

6. Use of Object-Oriented Programming (OOP)

Our Flight Booking Management System extensively utilizes Object-Oriented Programming (OOP) principles to create a robust and maintainable codebase. Here's how we applied OOP concepts:

- **Encapsulation**: We encapsulated the attributes of classes such as Flight, Passenger, Booking, and FlightTicketBookingSystem by making them private and providing public getter and setter methods. This ensures that the internal state of an object is protected from unauthorized access and modifications.
- **Inheritance**: The Flight class is a base class that provides common attributes and methods for all flights. We extended this class into InternationalFlight and DomesticFlight classes, which inherit the properties of the Flight class and add additional attributes specific to international and domestic flights, such as passportRequirement and idRequirement.
- **Polymorphism**: We used polymorphism to handle different types of flights (international and domestic) in a uniform way. For example, in the searchFlightByOriginAndDestination method, we check the type of flight using the instanceof keyword to provide appropriate handling for international and domestic flights.
- **Abstraction**: We abstracted the complex operations involved in booking and managing flights and passengers into their respective classes. The FlightTicketBookingSystem class provides high-level methods for managing flights and bookings, hiding the implementation details from the main application logic

Flight Class

Variables:

- String flightNumber: Unique identifier for the flight.
- String origin: Origin of the flight.
- String destination: Destination of the flight.
- String schedule: Schedule of the flight.
- int totalSeats: Total number of seats available.
- boolean[] seats: Array representing the availability of seats.

Functions:

- Constructors and getter/setter methods.
- getFlightDetails(): Returns the details of the flight.
- bookSeat(): Books a seat in the flight.
- cancelSeat(int seatNumber): Cancels a seat in the flight.
- getAvailableSeats(): Returns the number of available seats.

InternationalFlight Class (Extends Flight)

Variables:

- String passportRequirement: Passport requirement for the flight.

Functions:

- Constructor and getter/setter methods.
- Overrides getFlightDetails() to include passport requirement.

DomesticFlight Class (Extends Flight)

Variables:

- String idRequirement: ID requirement for the flight.

Functions:

- Constructor and getter/setter methods.
- Overrides getFlightDetails() to include ID requirement.

Passenger Class

Variables:

- String passengerID: Unique identifier for the passenger.
- String name: Name of the passenger.
- int age: Age of the passenger.
- String passportNumber: Passport number of the passenger.

Functions:

- Constructors and getter/setter methods.
- getPassengerDetails(): Returns the details of the passenger.

Booking Class

Variables:

- String bookingID: Unique identifier for the booking.
- String flightNumber: Flight number of the booked flight.
- String passengerID: Passenger ID of the passenger.
- int seatNumber: Seat number of the booking.
- String bookingStatus: Status of the booking.

Functions:

- Constructors and getter/setter methods.
- getBookingDetails(): Returns the details of the booking.

FlightTicketBookingSystem Class

Variables:

- Flight[] flights: Array of flights.
- Passenger[] passengers: Array of passengers.
- Booking[] bookings: Array of bookings.
- int flightCount: Number of flights added.
- int passengerCount: Number of passengers added.
- int bookingCount: Number of bookings made.

Functions:

- Constructors and getter/setter methods.
- addFlight(Flight flight): Adds a flight to the system.
- removeFlight(String flightNumber): Removes a flight from the system.
- addPassenger(Passenger passenger): Adds a passenger to the system.
- removePassenger(String passengerID): Removes a passenger from the system.
- getPassengers(): Returns the array of passengers.
- bookTicket(String flightNumber, Passenger passenger): Books a ticket for a flight.
- cancelTicket(String bookingID): Cancels a booking.
- searchFlight(String flightNumber): Searches for a flight by its number.
- searchFlightByOriginAndDestination(String origin, String destination, String flightType): Searches for flights by origin, destination, and flight type.
- searchPassenger(String passengerID): Searches for a passenger by their ID.
- generateReport(): Generates a report of all flights.
- generateBookingID(): Generates a unique booking ID.
- findAvailableSeat(Flight flight): Finds an available seat in a flight.
- restoreSeat(String flightNumber, int seatNumber): Restores a seat after a booking cancellation.