FLIGHT MANAGEMENT SYSTEM

PREPARED FOR

JEE Cloud Training Program

Capgemini Technology Services

PREPARED BY

Devang Singh

Navya Likhita

Surya

Abstract:

The Flight Management System is a Java-based booking solution for flight tickets. It consolidates data provided by different airline carriers and hence provides the user details and rates in real-time. Travellers may want to make changes in their bookings. The application allows them to book, cancel, view and update their bookings with ease. Other than this, it eases the management of bookings too. All the bookings, flights, schedules and routes can be viewed, added and modified on a single application by the administrator.

Scopes:

Inscope:
Following is the functionality provided by the system:

There are two categories of people who would access the system: customer and administrator. Each of these would have some exclusive privileges.

1. The customer can:
    a. Create his user account.
    b. Login into the application.
    c. Check for available flights.
    d. Make a booking.
    e. View the bookings made.
    f. Cancel or modify a booking.

2. The administrator can:
    a. Login into the application.
    b. Add flight, schedule and route details.
    c. View the flight, schedule and route details.
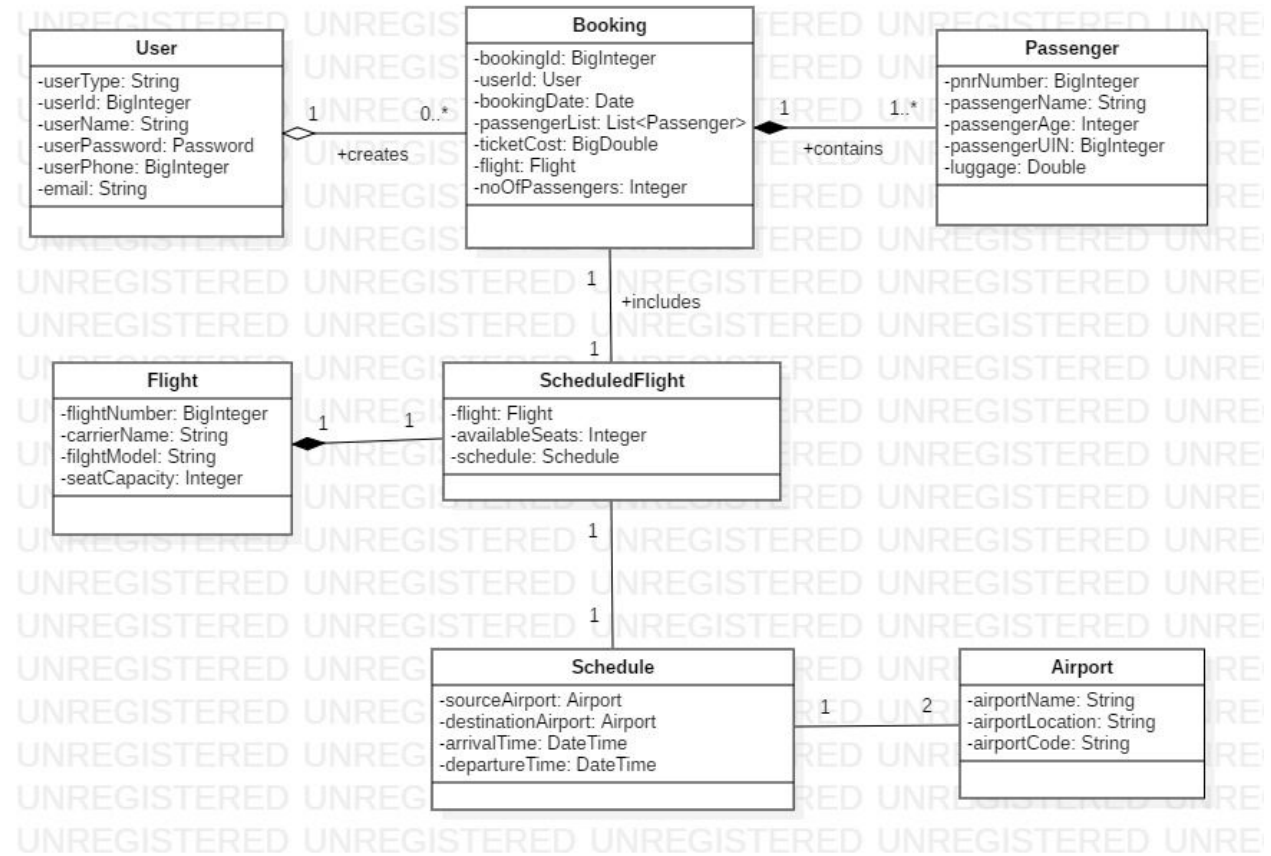   d. Cancel or modify the flight, schedule and route details.

Outscope:
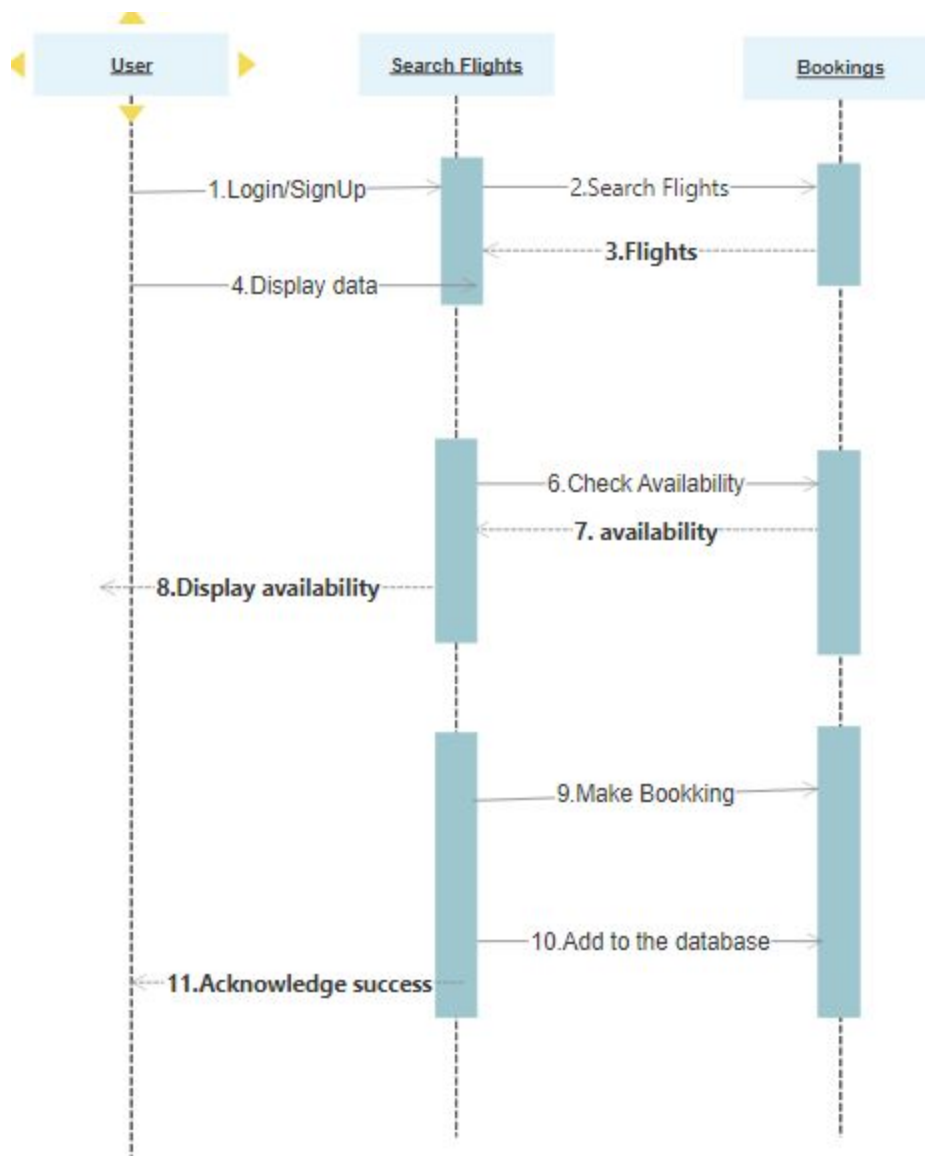The following functionalities have not been covered under the application:

1. The application does not cover boarding pass generation and seating plans.
2. Third party applications like email & sms integrations.
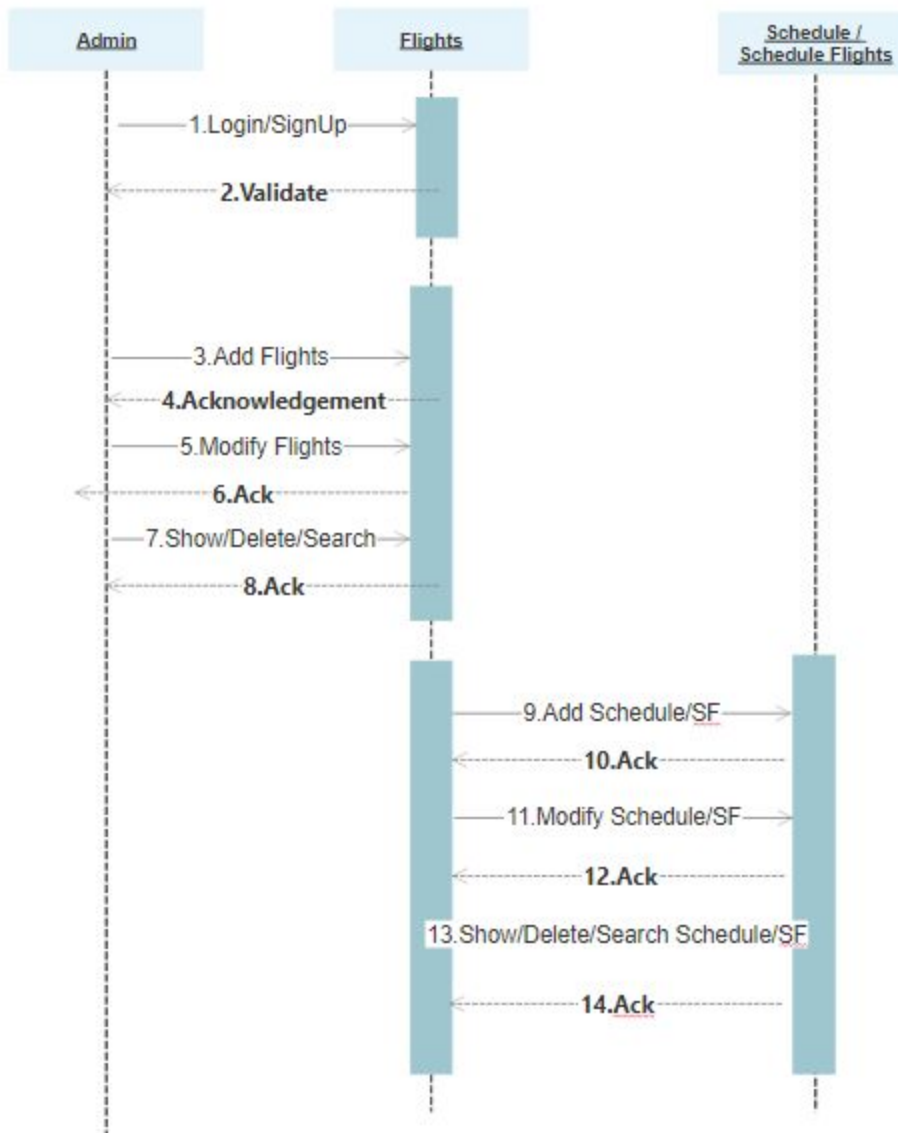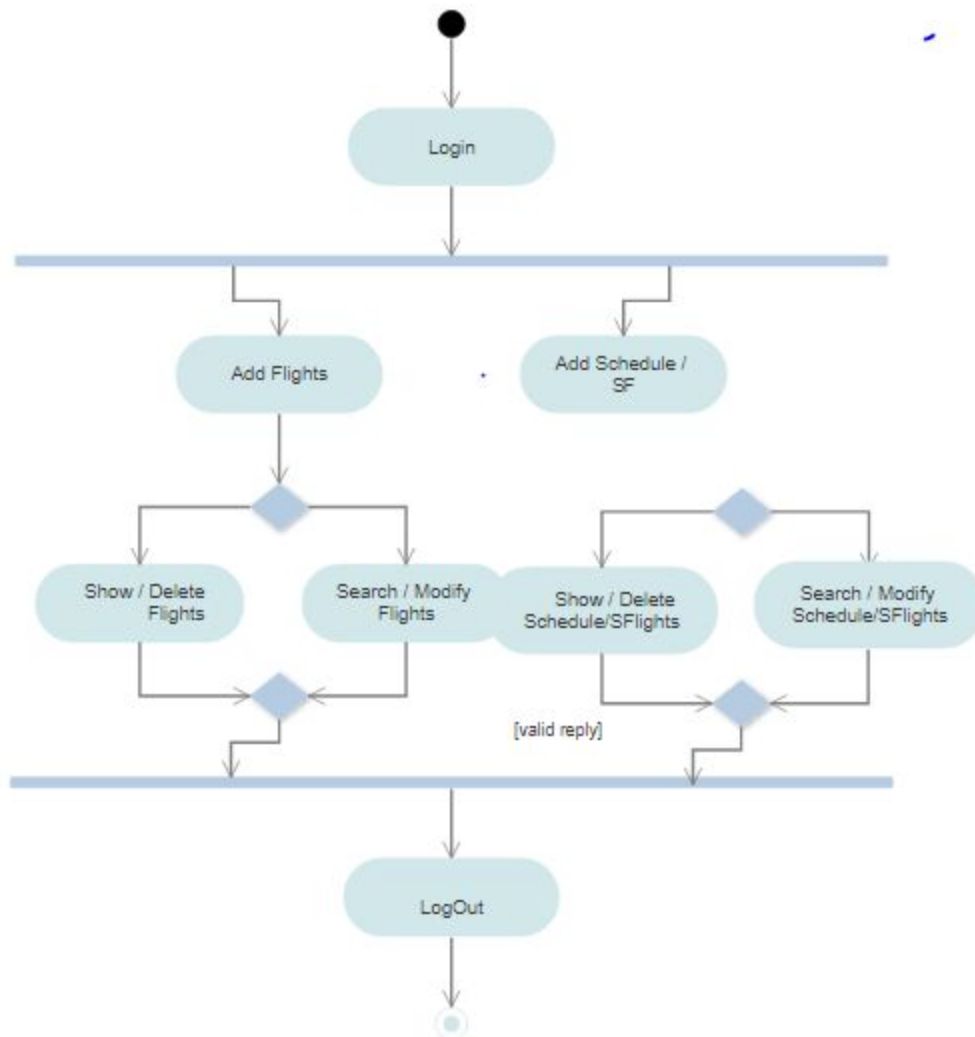3. Payments are not yet accepted by the application.
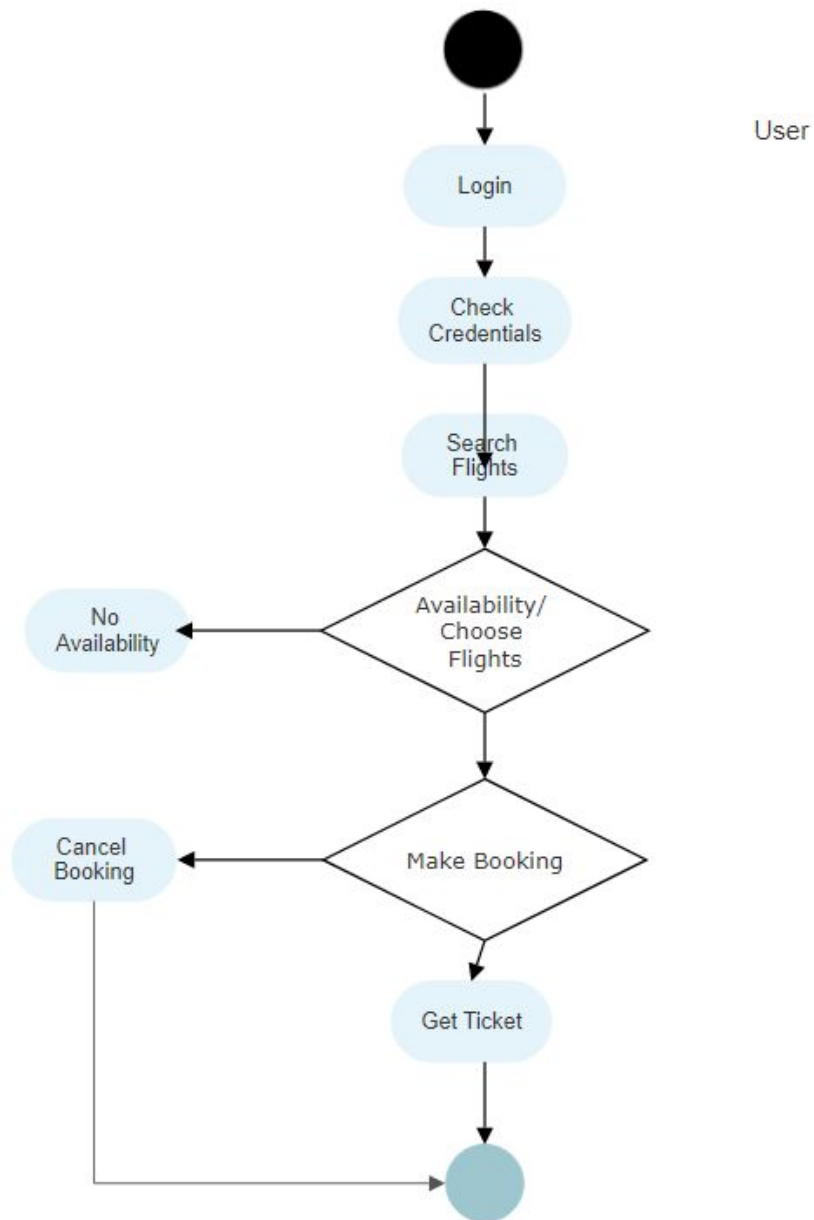
Class Diagram:

DTO Layer-

User
- -userType: String
- -userId: BigInteger
- -userName: String
- -userPassword: Password
- -userPhone: BigInteger
- -email: String

Booking
- -bookingId: BigInteger
- -userId: User
- -bookingDate: Date
- -passengerList: List<Passenger>
- -ticketCost: BigDouble
- -flight: Flight
- -noOfPassengers: Integer

Passenger
- -pnrNumber: BigInteger
- -passengerName: String
- -passengerAge: Integer
- -passengerUIN: BigInteger
- -luggage: Double

1   0..*
+creates

1   1..*
+contains

1
+includes

1

Flight
- -flightNumber: BigInteger
- -carrierName: String
- -filghtModel: String
- -seatCapacity: Integer

ScheduledFlight
- -flight: Flight
- -availableSeats: Integer
- -schedule: Schedule

1   1

1

1

Schedule
- -sourceAirport: Airport
- -destinationAirport: Airport
- -arrivalTime: DateTime
- -departureTime: DateTime

1   2

Airport
- -airportName: String
- -airportLocation: String
- -airportCode: String

Sequence Diagram for User:

User   **Search Flights**   **Bookings**

1.Login/SignUp → 2.Search Flights →

← 3.Flights

4.Display data

6.Check Availability →

← 7. availability

← 8.Display availability

9.Make Bookking →

10.Add to the database →

← 11.Acknowledge success

Sequence Diagram for Admin:

Admin | Flights | Schedule / Schedule Flights

1. Login/SignUp →
2. Validate ←
3. Add Flights →
4. Acknowledgement ←
5. Modify Flights →
6. Ack ←
7. Show/Delete/Search →
8. Ack ←
9. Add Schedule/SF →
10. Ack ←
11. Modify Schedule/SF →
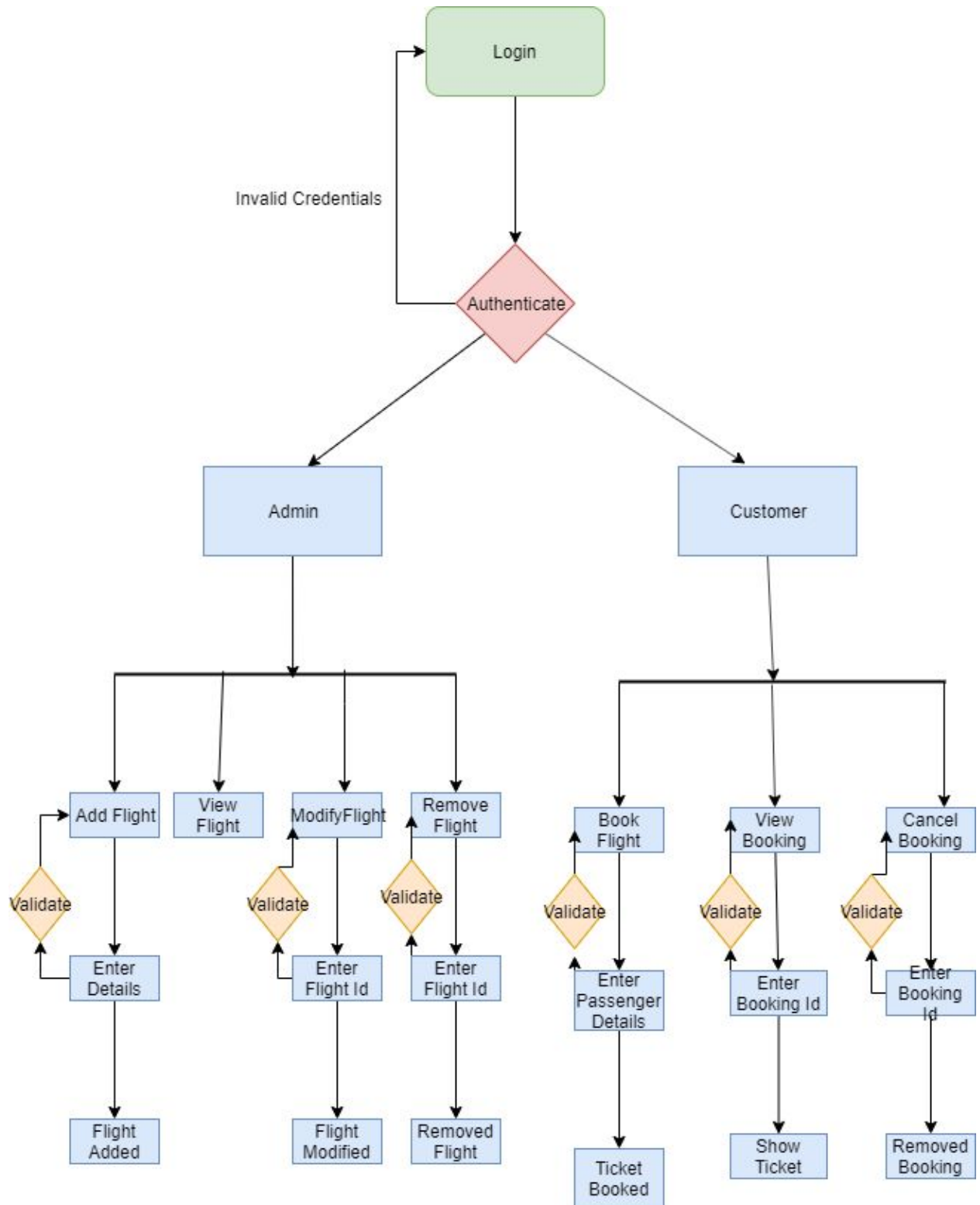12. Ack ←
13. Show/Delete/Search Schedule/SF
14. Ack ←

Activity Diagram for Admin:

Activity Diagram for User:
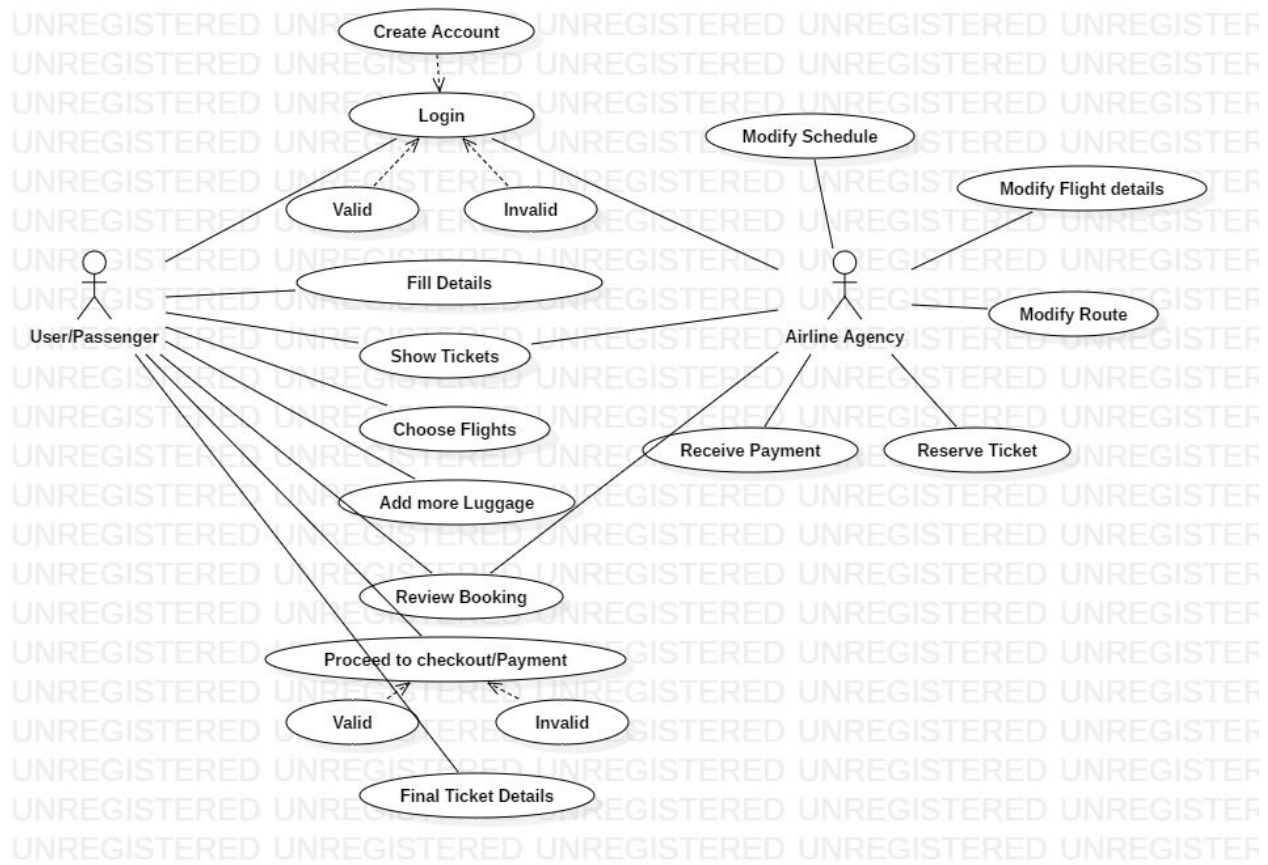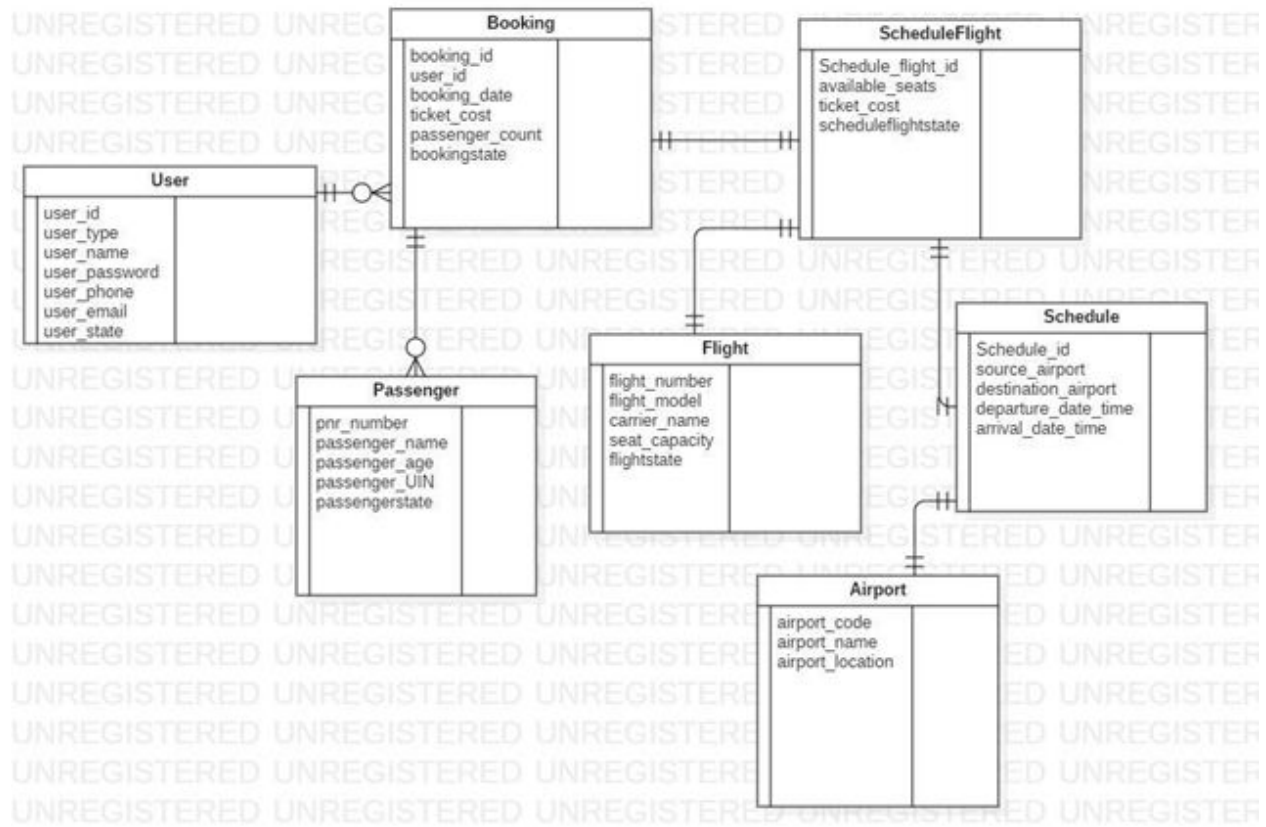
Flow Chart:

Use Case Diagram:

Entity Relation Diagram:

Booking
- booking_id
- user_id
- booking_date
- ticket_cost
- passenger_count
- bookingstate

ScheduleFlight
- Schedule_flight_id
- available_seats
- ticket_cost
- scheduleflightstate

User
- user_id
- user_type
- user_name
- user_password
- user_phone
- user_email
- user_state

Passenger
- pnr_number
- passenger_name
- passenger_age
- passenger_UIN
- passengerstate

Flight
- flight_number
- flight_model
- carrier_name
- seat_capacity
- flightstate

Schedule
- Schedule_id
- source_airport
- destination_airport
- departure_date_time
- arrival_date_time

Airport
- airport_code
- airport_name
- airport_location

Wireframes:

✈ Flight Reservation System   [Log Out]

Booking Details:

Passenger Details:

👤	Passenger Name	Age	Gender	ID Number
👤	Passenger Name	Age	Gender	ID Number
👤	Passenger Name	Age	Gender	ID Number
👤	Passenger Name	Age	Gender	ID Number

*Maximum 4 passengers per booking

Billing Address: Contact Number:

| 🏠 | Address | | 📞 | Contact Number |

*Booking Confirmation will be sent to the above mentioned contact number.

☑ I agree to the terms and conditions.

[Confirm Booking]

Flight Reservation System 2019   About Us

Your booking has been confirmed.

Booking Id: xxxxxxxxxx

Flight Details:
Number: xxxxxxxxxx
Source: xxxxxxxxxxxxxxxxxxxxxxxxxx, xxxxx
Destination: xxxxxxxxxxxxxxxxxxxxxxxxxx, xxxxx
Departure: xx-xx-xxxx xx:xx:xx
Arrival: xx-xx-xxxx xx:xx:xx

Passenger Details:

Passenger 1:
PNR: xxxxxxxxxxxx
Name: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Age: xx   Gender: xx   ID: xxxxxxxxxxxx

Passenger 2:
PNR: xxxxxxxxxxxx
Name: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Age: xx   Gender: xx   ID: xxxxxxxxxxxx

Passenger 4:
PNR: xxxxxxxxxxxx
Name: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Age: xx   Gender: xx   ID: xxxxxxxxxxxx

Passenger 3:
PNR: xxxxxxxxxxxx
Name: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Age: xx   Gender: xx   ID: xxxxxxxxxxxx

Contact Number: xxxxxxxxxx

Billing Address: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Download eTicket

Book Domestic and International Flights :

FROM:

TO:

ON:

| Source Airport | ▾ |
| Select |
| Airport Name (Code) |
| Airport Name (Code) |
| Airport Name (Code) |

| Destination Airport | ▾ |
| Select |
| Airport Name (Code) |
| Airport Name (Code) |
| Airport Name (Code) |

4/22/2012   📅▾

Search

Available Flights:

Flight Number: xxxxxxxxxxx
Carrier Name: xxxxxxxxxxxx
Flight Model: xxxxxxxxxxxx
Departure: xx-xx-xx xx:xx:xx
Arrival: xx-xx-xx xx:xx:xx
Cost/Seat: ₹ xx.xx

BOOK NOW

Flight Number: xxxxxxxxxxx
Carrier Name: xxxxxxxxxxxx
Flight Model: xxxxxxxxxxxx
Departure: xx-xx-xx xx:xx:xx
Arrival: xx-xx-xx xx:xx:xx
Cost/Seat: ₹ xx.xx

BOOK NOW

Flight Number: xxxxxxxxxxx
Carrier Name: xxxxxxxxxxxx
Flight Model: xxxxxxxxxxxx
Departure: xx-xx-xx xx:xx:xx
Arrival: xx-xx-xx xx:xx:xx
Cost/Seat: ₹ xx.xx

BOOK NOW

Flight Number: xxxxxxxxxxx
Carrier Name: xxxxxxxxxxxx
Flight Model: xxxxxxxxxxxx
Departure: xx-xx-xx xx:xx:xx
Arrival: xx-xx-xx xx:xx:xx
Cost/Seat: ₹ xx.xx

BOOK NOW

✈ Flight Reservation System

Book Domestic and International Flights :

FROM:

| Source Airport | ▾ |

| **Select** |
| Airport Name (Code) |
| Airport Name (Code) |
| Airport Name (Code) |

TO:

| Destination Airport | ▾ |

| **Select** |
| Airport Name (Code) |
| Airport Name (Code) |
| Airport Name (Code) |

ON:

| 4/22/2012 | 📅▾ |

Search

Check Out the Best Offers Available :

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla quam velit, vulputate eu pharetra nec, mattis ac neque. Duis vulputate commodo lectus, ac blandit elit tincidunt id. Sed rhoncus, tortor sed eleifend tristique, tortor mauris molestie elit, et lacinia ipsum quam nec dui. Quisque nec mauris sit amet elit iaculis pretium sit amet quis magna. Aenean velit odio, elementum in tempus ut, vehicula eu diam. Pellentesque rhoncus aliquam mattis. Ut vulputate eros sed felis sodales nec vulputate justo hendrerit. Vivamus varius pretium ligula, a aliquam odio euismod sit amet. Quisque laoreet sem sit amet orci ullamcorper at ultricies metus viverra. Pellentesque arcu mauris, malesuada quis ornare accumsan, blandit sed diam.

Flight Reservation System 2019   About Us

Your bookings:

Booking Id: xxxxxxxxx

Upcoming

Flight Details:
Number: xxxxxxxxx
Source: xxxxxxxxxxxxxxxxxxxxxxxxx, xxxxx
Destination: xxxxxxxxxxxxxxxxxxxxxxxxx, xxxxx
Departure: xx-xx-xxxx xx:xx:xx
Arrival: xx-xx-xxxx xx:xx:xx

Passenger Details:

Passenger 1:
PNR: xxxxxxxxxxxx
Name: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Age: xx   Gender: xx   ID: xxxxxxxxxxxx

Passenger 2:
PNR: xxxxxxxxxxxx
Name: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Age: xx   Gender: xx   ID: xxxxxxxxxxxx

Passenger 4:
PNR: xxxxxxxxxxxx
Name: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Age: xx   Gender: xx   ID: xxxxxxxxxxxx

Passenger 3:
PNR: xxxxxxxxxxxx
Name: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Age: xx   Gender: xx   ID: xxxxxxxxxxxx

Contact Number: xxxxxxxxx

Billing Address: xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Cancel   Download eTicket

Booking Id: xxxxxxxxx

Cancelled

Flight Details:
Number: xxxxxxxxx

Flight Reservation System   LOG OUT

Welcome User!!!

Book A Flight

View Available Flights

Cancel A Booking

Flight Reservation System 2019   About Us

✈ Flight Reservation System

LOG OUT

Add	View	Search	Update

Flight Id: xxxx

Flight Model: xxxx

Carrier Name: xxxx

Seat Capacity: xxxx

Update **Remove**

Flight Id: xxxx

Flight Model: xxxx

Carrier Name: xxxx

Seat Capacity: xxxx

Update **Remove**

Flight Reservation System 2019   About Us

✈ Flight Reservation System

Add	View	Search	Update

🔍 Flight Id: xxxx

Flight Id: xxxxx

Flight Model: xxxxx

Carrier Name: xxxxx

Seat Capacity: xxxxxx

Update   **Remove**

Flight Reservation System

Add	View	Search	Update

Flight Id: xxxx

Flight Model: xxxx

Carrier Name: xxxx

Seat Capacity: xxxxx

Update

UserName :   [   ]

Password :   [   ]

Confirm Password : [   ]

DOB :   [/ /   ] 📅

                    [SignUp]

Flight Reservation System   LOGOUT

Add Schedule Flights:

Schedule Flight Id : [_____]

Source Airport : [_____]

Destination Airport : [_____]

Departure Time : [/ /____] 📅

Arrival Time : [/ /____] 📅

Ticket Cost : [_____]

[Add]

Flight Reservation System 2019   About Us

Delete Schedule Flight :

Enter Schedule FlightId : | xxxxx | [Delete]

Alert

The schedule lfight has been
removed.

UserName :   [   ]

Password :   [   ]

[Login]

Flight Reservation System   LOGOUT

Modify Schedule Flights:

Enter Schedule Flight Id : [   ]   [Search]

Details :

Source Airport :   [   ]

Destination Airport :   [   ]

Departure Time :   [/ /   ] 📅

Arrival Time :   [/ /   ] 📅

Ticket Cost :   [   ]

                                                              [Modify]

Flight Reservation System 2019   About Us

Search Schedule Flights :

Enter Schedule Flight Id :   Search

Details :

Source Airport :

Destination Airport :

Departure Time :   / /   📅

Arrival Time :   / /   📅

Ticket Cost :

Class and Method Description:

DTO Layer:

1. **User**: This class stores the user type (admin or the customer) and all user information.

   Attributes:
   - userType: String.
   - userId: BigInteger
   - userName: String
   - userPassword: Password
   - userPhone:BigInteger
   - userEmail: String

   Methods: –

2. **Passenger**: This class stores all the details of the travelling passenger.

   Attributes:
   - pnrNumber: BigInteger
   - passengerName: String
   - passengerAge: Integer
   - passengerUIN: BigInteger
   - Luggage: Double

   Methods: –

3. **Booking**: This class stores the details of a booking made by a particular userId. Every booking stores a list of passengers travelling in it as well as the flight details.

   Attributes:
   - bookingId: BigInteger
   - userId: User
   - bookingDate: Date
   - passengerList: List<Passenger>
   - ticketCost: BigDouble
   - flight: Flight
   - noOfPassengers: Integer

   Methods: –

4. **ScheduledFlight**: This class stores a flight that is scheduled along with its schedule and the vacancy.

Attributes:
- flight: Flight
- availableSeats: Integer
- schedule: Schedule

Methods: -

5. **Flight**: This class stores all the details of a flight.

Attributes:
- flightNumber: BigInteger
- flightModel: String
- carrierName: String
- seatCapacity: Integer

Methods: -

6. **Schedule**: This class stores a flight schedule.

Attributes:
- sourceAirport: Airport
- destinationAirport: Airport
- arrivalTime: DateTime
- departureTime: DateTime

Methods: -

7. **Airport**: This class stores the details of an airport.

Attributes:
- airportName: String
- airportCode: String
- airportLocation: String

Methods: -

Service Layer:

8. **UserServiceImpl**:

Attributes: -

Methods:
- addUser(User):User :- Adds a new user.

viewUser(BigInteger):User :-
Shows the details of a user identifiable by the user id.

viewUser(): List<User> :-
Shows the details of all users.

updateUser(User):User :-
Updates the details of a user.

deleteUser(BigInteger):void
Removes a user as per the user id.

validateUser(User): void :-
Validates the attributes of a user.

9. **BookingServiceImpl:**
   Attributes: -
   Methods:
   addBooking(Booking):Booking :- Creates a new booking.

   modifyBooking(Booking): Booking :- Modifies a previous booking. All information related to the booking except the booking id can be modified.

   viewBooking(BigInteger): List<Booking> :- Retrieves a booking made by the user based on the booking id.

   viewBooking(): List<Booking> :- Retrieves a list of all the bookings made.

   deleteBooking(BigInteger): void :-
   Deletes a previous booking identifiable by the 'bookingId'.

   validateBooking(Booking): void :-
   Validates the attributes of a booking.

   validatePassenger(Passenger): void :-
   Validates the attributes of a passenger.

10. **FlightServiceImpl:**
    Attributes: -
    Methods:
    addFlight(Flight): Flight :-

Adds a new flight which can be scheduled.

modifyFlight(Flight): Flight :-
Modify the details of a flight.

viewFlight(BigInteger): Flight :-
Shows the details of a flight specified by the flight number.

viewFlight(): List<Flight> :-
View the details of all flights.

deleteFlight(BigInteger): void :-
Removes a flight.

validateFlight(Flight): void :-
Validates the attributes of a flight.

11. **ScheduleFlightServicesImpl:**

Attributes: -
Methods:

scheduleFlight(ScheduledFlight): ScheduledFlight :-
Schedules a flight alongwith its timings, locations and capacity

viewScheduledFlights(Airport, Airport, LocalDate): List<Scheduled Flight> :-
Returns a list of flights between two airports on a specified date.

viewScheduledFlights(BigInteger):Flight :-
Returns a list of a scheduled flight identifiable by flight number.

viewScheduledFlight(): List<ScheduledFlight> :-
Shows all the details and status of all flights.

modifyScheduledFlight(Flight,Schedule, Integer): ScheduledFlight :-
Modifies the details of a scheduled flight.

deleteScheduledFlight(BigInteger): void :-
Removes a flight from the available flights.

validateScheduledFlight(ScheduledFlight): void :-
Validates the attributes of a scheduled Flight.

12. **AirportServiceImpl:**

Attributes: -
Methods :
> viewAirport(): List<Airport> :-
> Returns the list of all airports.
>
> viewAirport(String): Airport :-
> Returns the details of an airport identifiable by the airport code.

DAO Layer:

13. UserDaoImpl:
> Attributes:
>> userList: List<User>
>
> Methods:
>> addUser(User):User :-
>> Adds a new user.
>>
>> viewUser(BigInteger):User :-
>> Shows the details of a user identifiable by the user id.
>>
>> viewUser(): List<User> :-
>> Shows the details of all users.
>>
>> updateUser(User):User :-
>> Updates the details of a user.
>>
>> deleteUser(BigInteger):void
>> Removes a user as per the user id.

14. BookingDaoImpl:
> Attributes:
>> bookingList: List<Booking>
>
> Methods:
>> *addBooking(Booking):Booking* :- Creates a new booking.

modifyBooking(Booking): Booking :- Modifies a previous booking. All information related to the booking except the booking id can be modified.

viewBooking(BigInteger): List<Booking> :- Retrieves a booking made by the user based on the booking id.

viewBooking(): List<Booking> :- Retrieves a list of all the bookings made.

deleteBooking(BigInteger): void :-
Deletes a previous booking identifiable by the 'bookingId'.

15. **FlightDaoImpl:**
        Attributes:
                flightList: List<Flight>

        Methods:
                addFlight(Flight): Flight :-
                Adds a new flight which can be scheduled.

                modifyFlight(Flight): Flight :-
                Modify the details of a flight.

                viewFlight(BigInteger): Flight :-
                Shows the details of a flight specified by the flight number.

                viewFlight(): List<Flight> :-
                View the details of all flights.

                deleteFlight(BigInteger): void :-
                Removes a flight.

16. **ScheduledFlightDaoImpl:**
        Attributes:
                scheduledFlightList: List<ScheduledFlight>

        Methods:
                scheduleFlight(ScheduledFlight): ScheduledFlight :-
                Schedules a flight alongwith its timings, locations and capacity

viewScheduledFlights(Airport, Airport, LocalDate): List<Scheduled Flight> :-
Returns a list of flights between two airports on a specified date.

viewScheduledFlights(BigInteger):Flight :-
Returns a list of a scheduled flight identifiable by flight number.

viewScheduledFlight(): List<ScheduledFlight> :-
Shows all the details and status of all flights.

modifyScheduledFlight(Flight,Schedule,int): ScheduledFlight :-
Modifies the details of a scheduled flight.

deleteScheduledFlight(BigInteger): void :-
Removes a flight from the available flights.

17. **AirportDaoImpl:**
Attributes:
airportList: List<Airport>

Methods:
viewAirport(): List<Airport> :-
Returns the list of all airports.

viewAirport(String): Airport :-
Returns the details of an airport identifiable by the airport code.

Validations:

1. The 'userPhone' should have an exact 10 digit number and the number should not start with zero.
2. Date and Time should be valid i.e date and time that has already elapsed shouldn't be entered
3. 'noOfPassenger' should always be less than equal to that of available seats.
4. The local part of the email should contain alphanumeric characters only. No special characters are to be present as the first character of the id.
5. The chosen airport's name should be present inside the Airport database.
6. The Unique Identification Number should be of 12 digits.

Assumptions:

However, we have made a few assumptions with respect to the application, which are:
1. Administrator and customer are both Users. They are differentiated by a variable 'userType' in the User class.
2. Every passenger needs to enter a Unique Identification Number while booking is being made. For simplicity, we assume it to be a 12-digit Aadhaar Number.
3. All flights are direct flights.
4. No flight gets cancelled.
5. Number of airports is fixed and stored in database.
6. All the flights are considered to be domestic.