DESIGN DOCUMENT

Part A - System Analysis:

A1 - System’s Description:

Background & Description

The Airline Flight Reservation System is a web-based application designed to facilitate seamless and efficient flight booking processes for various users, including Tourism Agents, Airline Agents, System Admins, and Registered Users. The system provides a user-friendly interface with functionalities tailored to meet the diverse needs of different stakeholders within the airline industry.

Functional Requirements for each user group:

Users

1. User Management
    a. Users can register
2. Flight Booking
    a. Users can search and select flights based on origin, destination and date
    b. Users booking flights will be able to select their seat as well as view the seat map.
    c. Users will be offered dynamic pricing for the seat selection process.
    d. Users can cancel their flight
3. Flight Management
    a. Users can manage their flight and see their information
4. Payment Processing
    a. Users can make secure payment for their flight
    b. Users can receive their ticket and receipt
5. Insurance Options
    a. Users can purchase flight cancellation insurance

Registered Users (same requirements as Users plus additional below)

1. User Management
    a. Registered users will be authenticated
2. Flight Booking
    a. Registered Users will opt for monthly promotions
    b. Registered Users will be discounted for using airport lounges
    c. Registered Users will be able to receive free companion tickets

Airline Agent (same requirements as Users plus additional below)

1. User Management
    a. Registered users will be authenticated
2. Bookings Management.
    a. Airline Agents can Browse Passengers on their flights

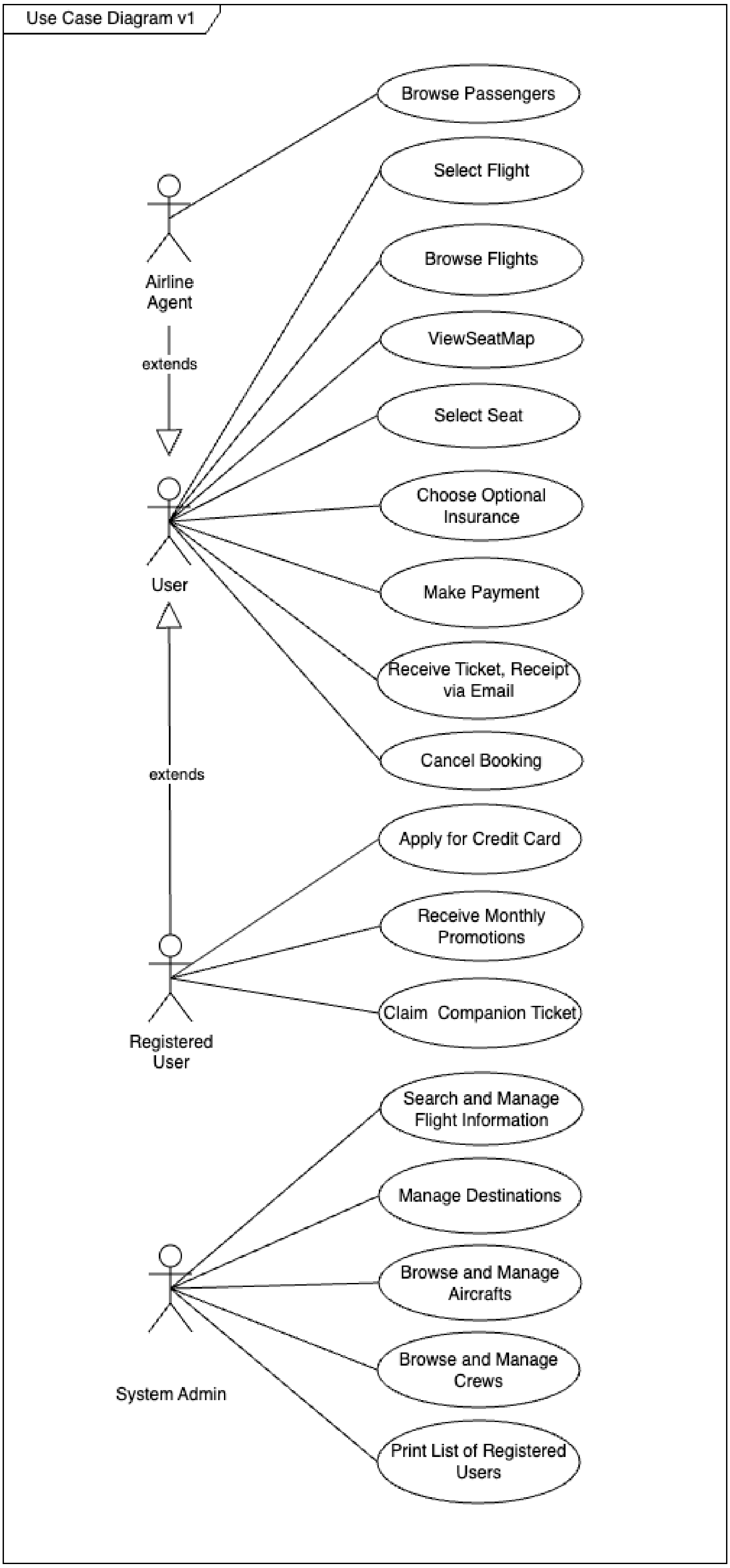
System Admins

1. User Management
    a. Admin users will be authenticated
2. Flight Management
    a. System Admins can search for flights
    b. System Admins can add, remove and modify flight information
    c. System Admins can manage flight destinations
3. Crew and Aircraft Management
    a. System Admins can search for crews and aircraft
    b. System Admins can add, remove and modify crew information
    c. System Admins can add, remove and modify aircraft information
4. Other
    a. View Registered User meta data such as print list of registered users

Non-Functional Requirements

1. Security (secure login, payment transactions)
2. User-friendly UI
3. Scalable system

A2 - System’s Use-Case Diagram:



A3 - System’s Scenarios for each Use-Case:

Use Case: Register
Participating actor(s): John (user)
Scenario: John selects the option “Member Login/Register” in the Flight Reservation System App. John sees inputs for “Email: ”, “Password: ” and an option for “ Register”. John chooses to register. The system provides input boxes for John to enter the following: “First Name:”, “Last Name:”, “Address:”, “Date of Birth:” “Email:” and “Password:” John fills in his personal information. John selects “Register” and the system validates his inputs have no errors, and stores his information in the database. The system notifies John of his registration with “Successfully Registered!” and sends him a confirmation email.

Use Case: User Login
Participating actor(s): Mark (registered user)
Scenario: Mark selects the option “Member Login/Register” in the Flight Reservation System App. Mark sees inputs for “Email: ”, “Password: ”. He enters his login information. The system authenticates his username and password, bringing him to his User Profile page.

Use Case: Claim Companion Ticket
Participating actor(s): Mark (registered user)
Scenario: Mark logs into his account and is on his User Profile. Under options, Mark clicks “Claim Companion Ticket”. The system confirms Mark has not already claimed a companion ticket in the past year and notifies Mark, “You Have Successfully Claimed Your Companion Ticket!”.

Use Case: Receive Monthly Promotional News
Participating actor(s): Mark (registered user)
Scenario: Mark logs into his account and is on his User Profile. Under options, Mark clicks “Opt-in For Monthly Promotions”. The system confirms Mark is not already signed up and adds him as an observer. Mark gets notified by “You Have Subscribed for the Monthly Promotions!.

Use Case: Apply for Credit Card
Participating actor(s): Mark (registered user)
Scenario: Mark logs into his account and is on his User Profile. Under options, Mark clicks “Apply For Airline Credit Card”. The system confirms Mark is not already signed up for the airline credit card and sends a request to the issuing bank. Mark gets notified by “You Have Successfully Applied for the Credit Card!”.

Use Case: Browse Flight
Participating actor(s): John (user)
Scenario: John opens the Flight Reservation App and under the “Search Flights” section John sees inputs for “Departure Airport ”, “Arrival Airport”, “Date” and a button for “Search Flights”. John inputs “YYC”, “YYZ” and “2023-12-20” respectively and clicks “Search Flights”. The airline’s “Available Flights” is updated with a list of 3 flight’s, showing the flight number, departure city, arrival city, and date corresponding to John’s search.

Use Case: Select Flight and Create Booking
Participating actor(s): John (user)
Scenario: John has searched for a flight and found an available flight he is interested in. John confirms the departure city, arrival city, and date are what he wants and clicks "Select Flight" for flight number "F123" on the list and the system initiates the booking process, bringing John to the add-ons page.

Use Case: Browse Seats, Select Seat and Insurance.
Participating actor(s): John (user)
Scenario: John is on the seat selection page. He sees a seat map of the flight and the number of available seats left in each class. John can click on the seats he would like to reserve in each class. The estimated total is displayed at the bottom of the page. John selects seat "18C" in the "Comfort Seat" section and the total is displayed as $334.00. John sees a checkbox to add cancellation insurance at an additional cost of $50. John checks the box and the system updates his total to $384.00.

Use Case: Make Payment, Receive Ticket and Receipt
Participating actor(s): John (user)
Scenario: John has selected a flight, his desired seat and his insurance add-on. John clicks "Confirm and Pay". A new page is loaded for John to enter his personal information "First Name:" , "Last Name:", "Email:" as well as his financial information - "Credit Card Number:", "Expiry Date", and "CVV:". John enters all his personal information and clicks the "Confirm and Pay" button to initiate a payment. The system validates John's personal information, while John's bank validates his payment request. John waits for a response and then is notified that the payment is successful with a prompt "Congratulations! Your flight is booked. You will receive your tickets and order receipt via email shortly." The system then automatically sends an email to John providing his flight ticket(s) and payment receipt. John receives the email, having his ticket including his info and "Flight #", "Seat #", and "Booking ID".

Use Case: Cancel Flight
Participating actor(s): John (user)
Scenario: John has already purchased a flight and can no longer make it. He opens the Airline Reservation App and clicks "Manage Existing Booking". He sees an input for "Booking ID". John checks his email containing his flight ticket for his Booking ID and enters it into the system. John selects "Search" and his booking appears along with an action to "Cancel Booking". John selects"Cancel Booking". The system removes his ticket, updates the availability of the seat, and sends a request to John's bank for a refund on his payment. John waits for a response and is notified "Cancellation Confirmed!" and receives a confirmation email.

Use Case: Browse and Modify Flight Information
Participating actor(s): Bob (system administrator)
Scenario: Bob logs in through the "Admin Login" page.
Bob wants to change some details about flight "FL123". Bob can browse the list of "All Flights" or can "Search" for his flight. Bob enters the flight into the "Search Flights" section and clicks "Search". Bob see's the flight appear along with "Change" buttons for the flights "Departure City", "Arrival City", "Date" and "Crew". Bob selects "Change" for Departure City and changes "Calgary" to "Edmonton".

Use Case: Browse and Manage Aircrafts
Participating actor(s): Bob (system administrator)
Scenario: Bob logs in through the “Admin Login” page.
Bob wants to add an aircraft to the database. Under the “Options” header, Bob see’s two options for managing aircraft, “Add Plane” and “Remove Plane”. Bob selects “Add Plane” and the page is updated with a new header “Add Plane” where Bob can input “ ID:” and “Type:”. Bob enters “006” and “Boeing-737” respectively and clicks “Add”. The system confirms the aircraft ID is available, adds the new aircraft to the database, and updates Bob with “The update has been confirmed”.

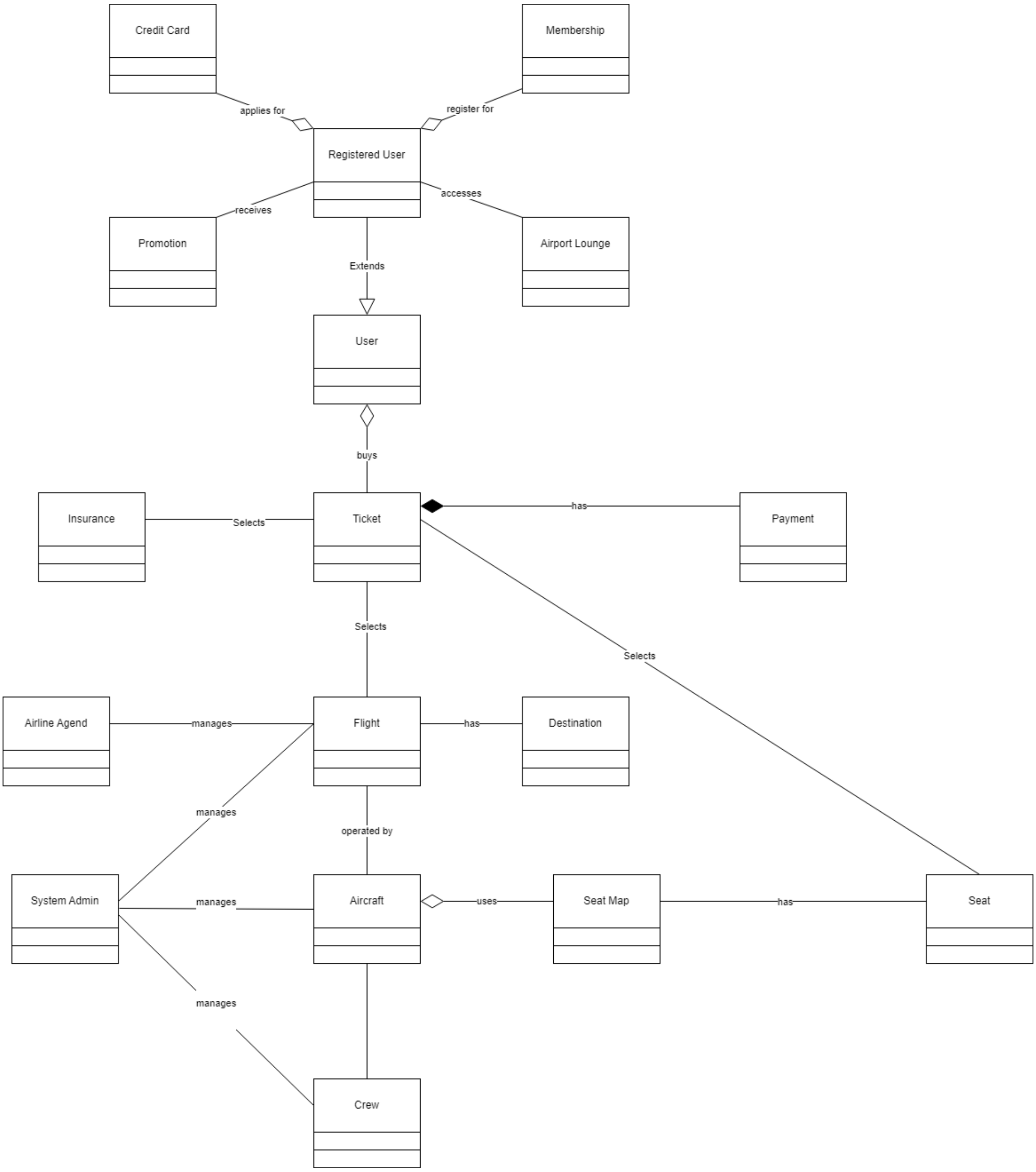
Use Case: Browse and Manage Crew
Participating actor(s): Bob (system administrator)
Scenario: Bob logs in through the “Admin Login” page.
Bob wants to check the crew assigned to a certain flight, and also add a crew to the database. Under “Search Flights”, Bob enters “F123” and clicks “Search”. Bob see’s the flight appear along with the crew ID assigned. Bob is content with this Crew. He then goes to the “Options” header, Bob see’s two options for managing crews, “Add Crew” and “Remove Crew”. Bob selects “Add Crew” and the page is updated with a new header “Add Crew” where Bob can input “ ID:” . Bob enters “E” and clicks “Add”. The system confirms the crew ID is available, adds the new crew to the database, and updates Bob with “The update has been confirmed”.

Use Case: Manage Destinations
Participating actor(s): Bob (system administrator)
Scenario: Bob logs in through the “Admin Login” page.
Bob wants to add a new destination to the database. Under the “Options” header, Bob see’s two options for managing destinations, “Add Destination” and “Remove Destination”. Bob selects “Add Destination” and the page is updated with a new header “Add Destination” where Bob can input “New Destination:”. Bob enters “London” and clicks “Add”. The system confirms the destination does not already exist, adds the new destination to the database, and updates Bob with “The update has been confirmed”.

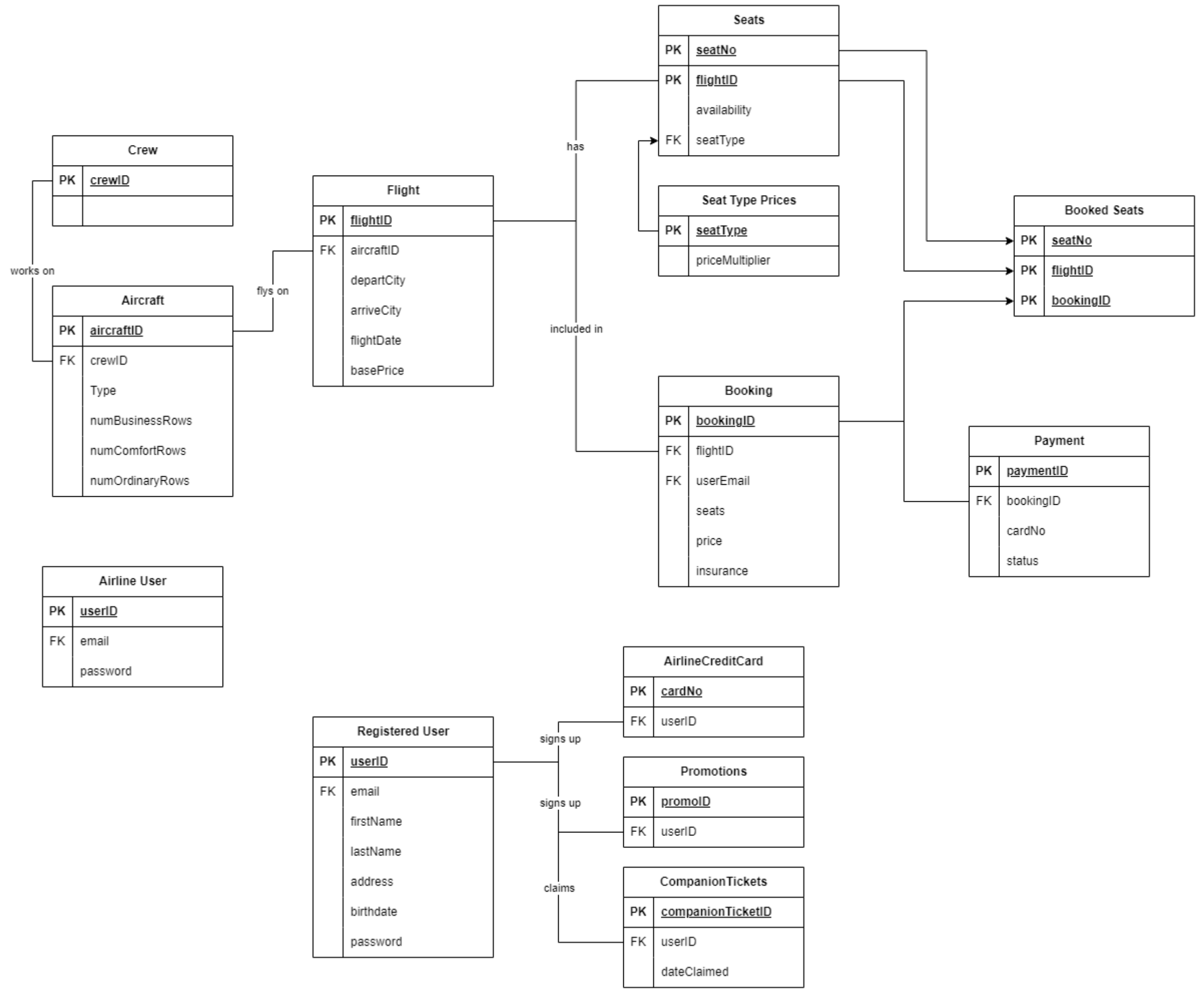
Use Case: Print Registered User List
Participating actor(s): Bob (system administrator)
Scenario: Bob logs in through the “Admin Login” page.
He needs to update his boss on the amount of registered users the company has. Bob goes to the “Options” header where there is a “Print Registered User List” button. Bob clicks it and is notified with “Printing List of Registered Users..”. Bob then see’s the Registered Users List containing each registered user along with their name and email.

Use Case: Browse Passenger List
Participating actor(s): Kate (airline agent)
Scenario: Kate logs in through the “Airline Agent Login” page.
She wants to browse all passengers for flight “F123”. Under “Flight Number: “, Kate enters the “F123” and clicks “Search”. The Passenger List displayed is populated with each passenger on the flight showing their “Name”, “email”, “BookingID” and “Seat”.

A4 - System's Conceptual Model:



Supplementary - ER Diagram:



Part B - Domain Diagrams:

B1 - Highlights of the system's architecture:

For our application, we are using a layered architecture style. In the presentation layer we have 4 different interfaces, User Interface, Registered User Interface, Airline Agent User Interface and Admin User Interface. These reflect the various portals and the views of the application each type of user gets. Each of these interfaces also have their own controllers which update the logic for these views. They handle all the dynamic changes of data and notify the view for change.

In the business layer we have 4 different managers, each with their own respective components. The Flight Reservation Manager has all the components for selecting, booking and canceling a flight. This includes any insurance and seat payment, as well as receipt handling. This is connected to all of the user types, except for the Admin, as they all should be able to use these features.

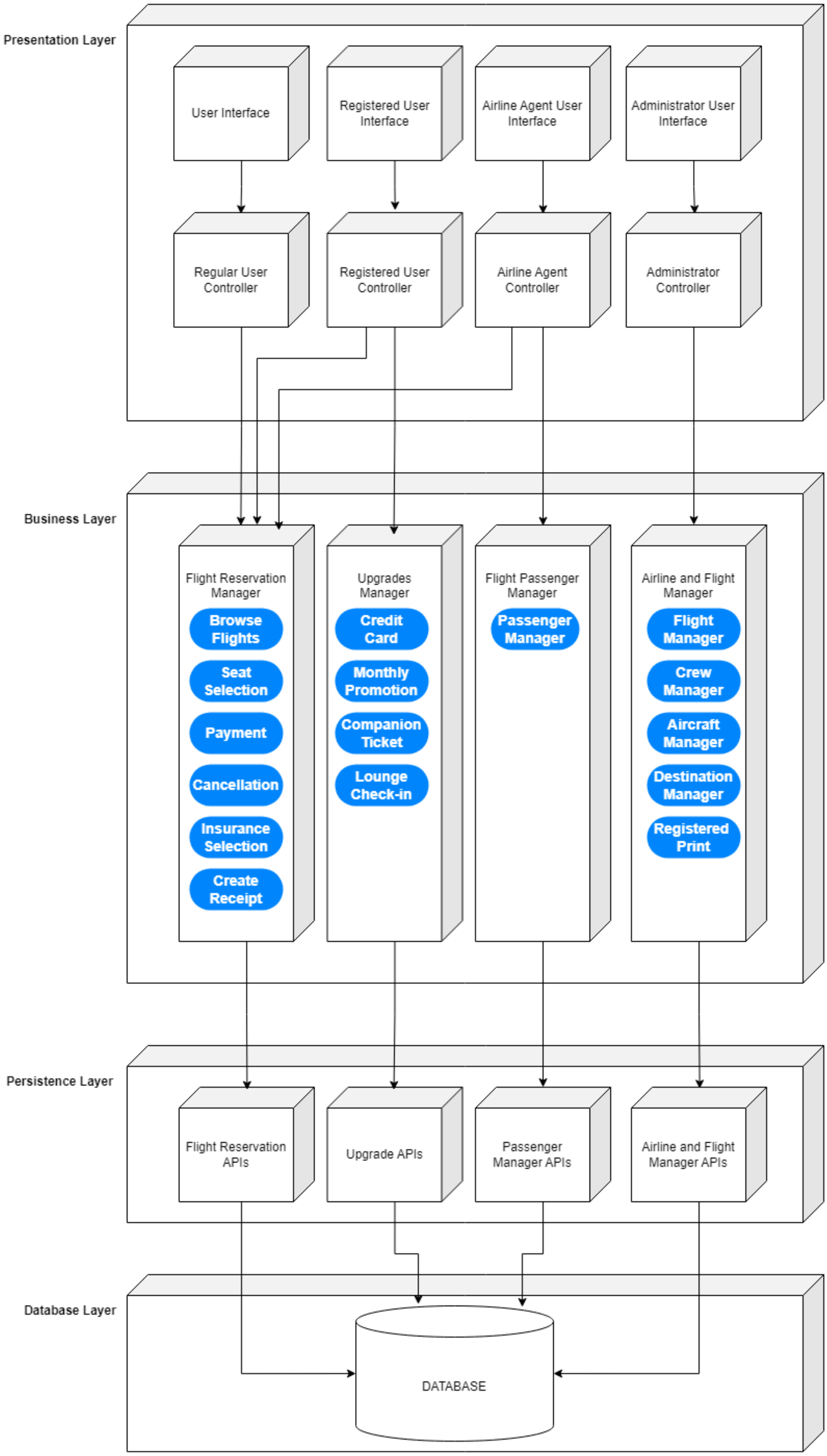
Next we have the Upgrades Manager which is responsible for the Credit Card component, Promotions, Companion Ticket and Check-in Lounge. This is connected to only the Registered User controller as they are controlled benefits for this user type.

Following this we have the Airline Agent Controller, which in addition to accessing features from Flight Reservation Manager, also has access to the Flight Passenger Manager. This allows the airline agent to Manage Passengers. In the context of our app, this allows the airline agent to view the list of passengers for a given flight and make any necessary changes such as removing users off a flight.

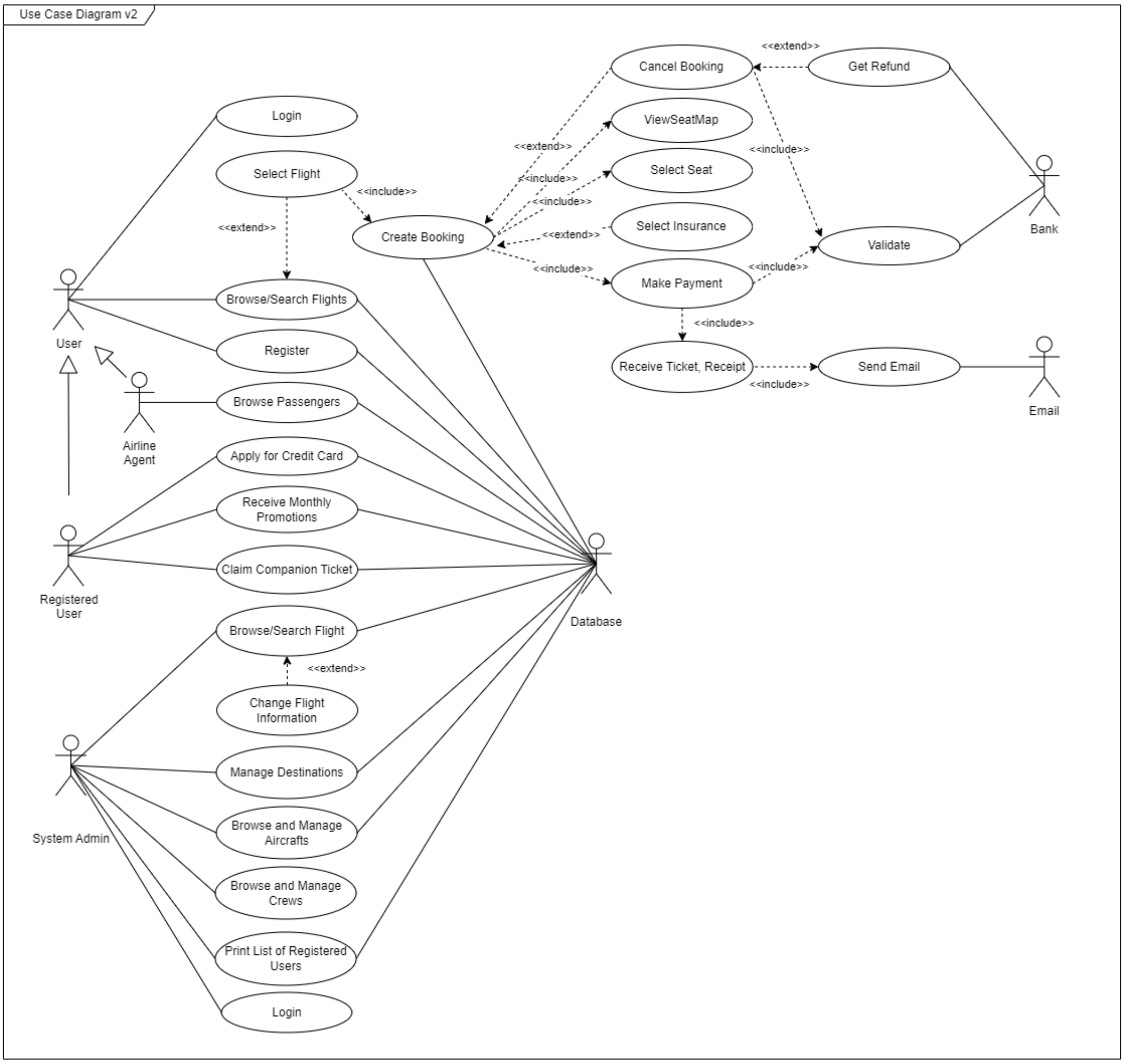
The Administrator controller is connected to the Airline Flight Manager which gives the admin the free access to manage flights, crews, aircrafts, destinations and also print the list of registered users.

Each of the four of these managers is connected to their respective API / Data objects in the Persistence layer. Ultimately all connecting to a single database that handles all the data transfer requests.

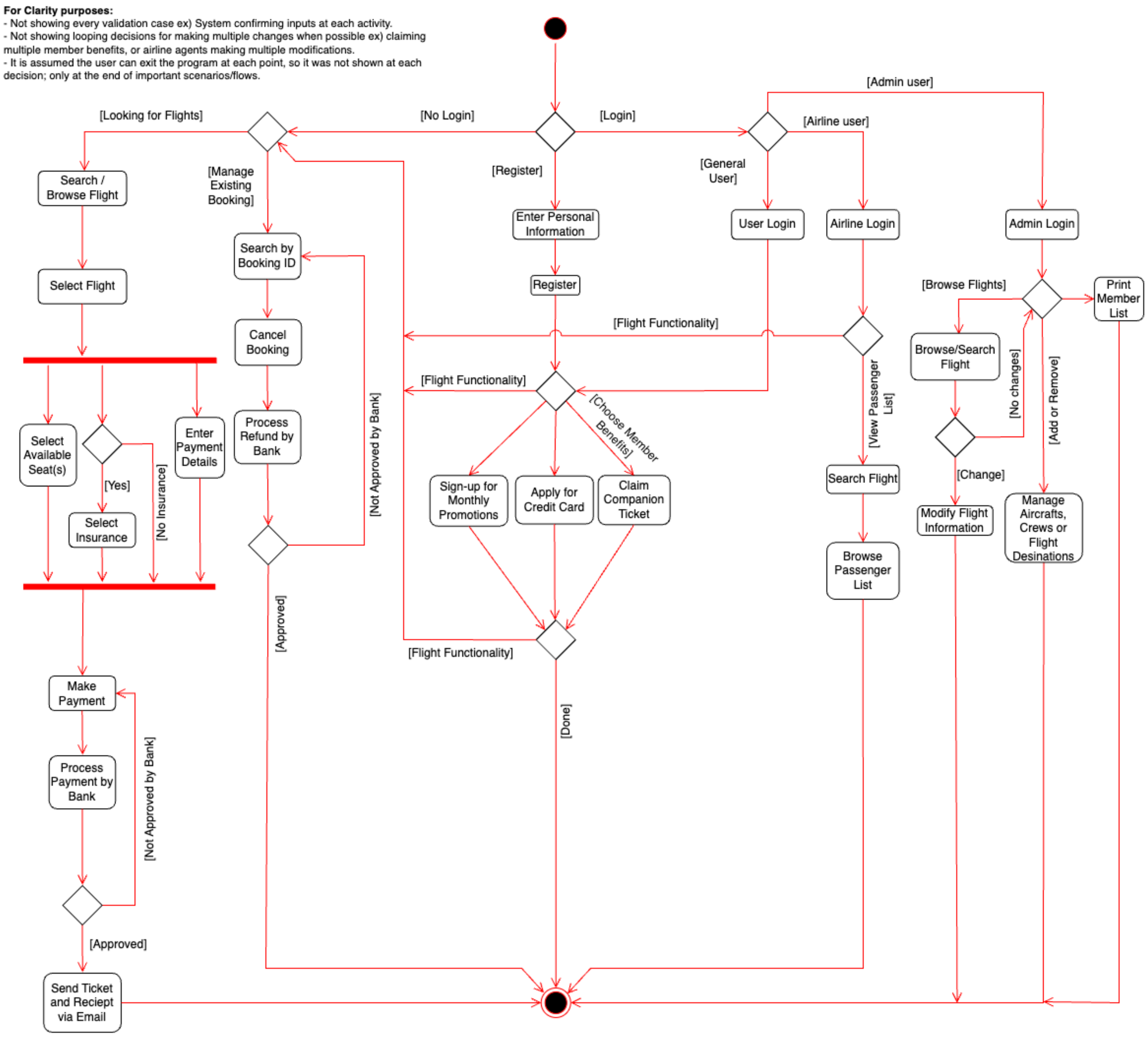
ENSF614 - Group 15



B2 - Updated Use-Case Diagram:



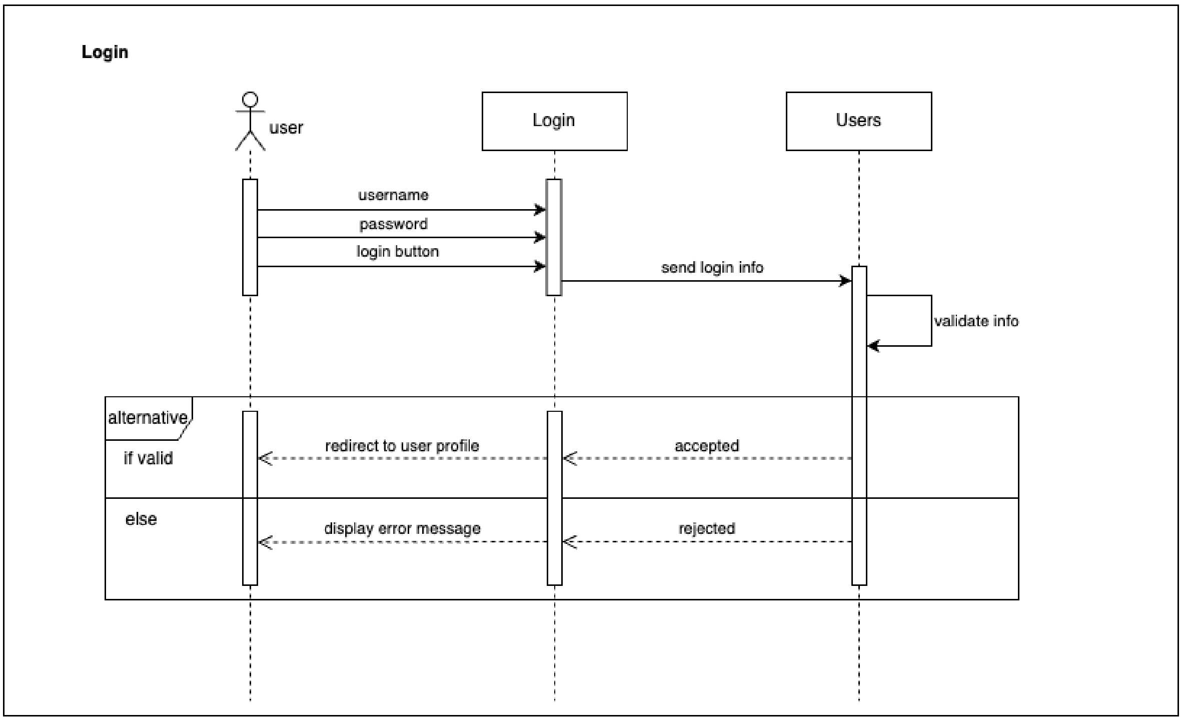
B3 - System’s Activity Diagram:

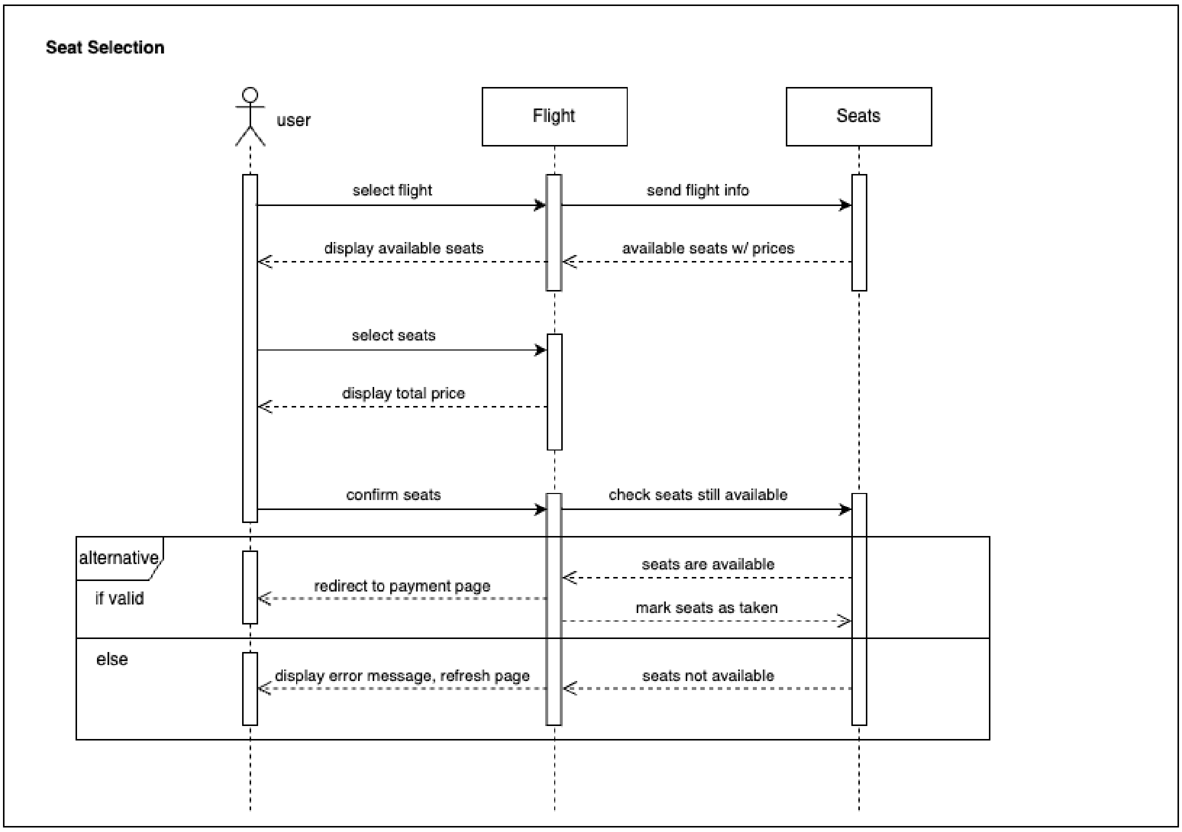


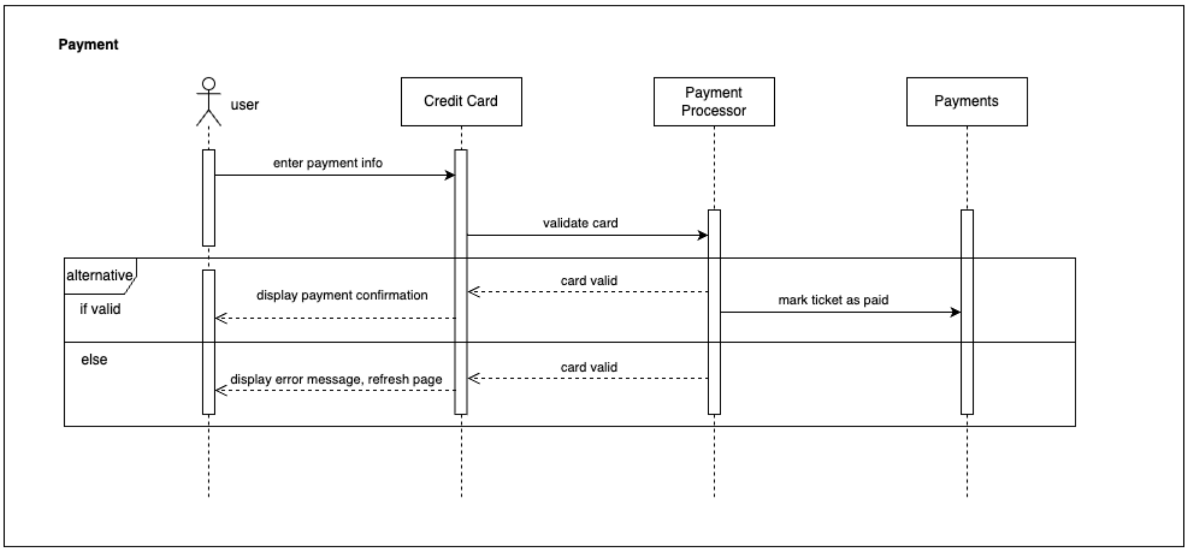
For Clarity purposes:

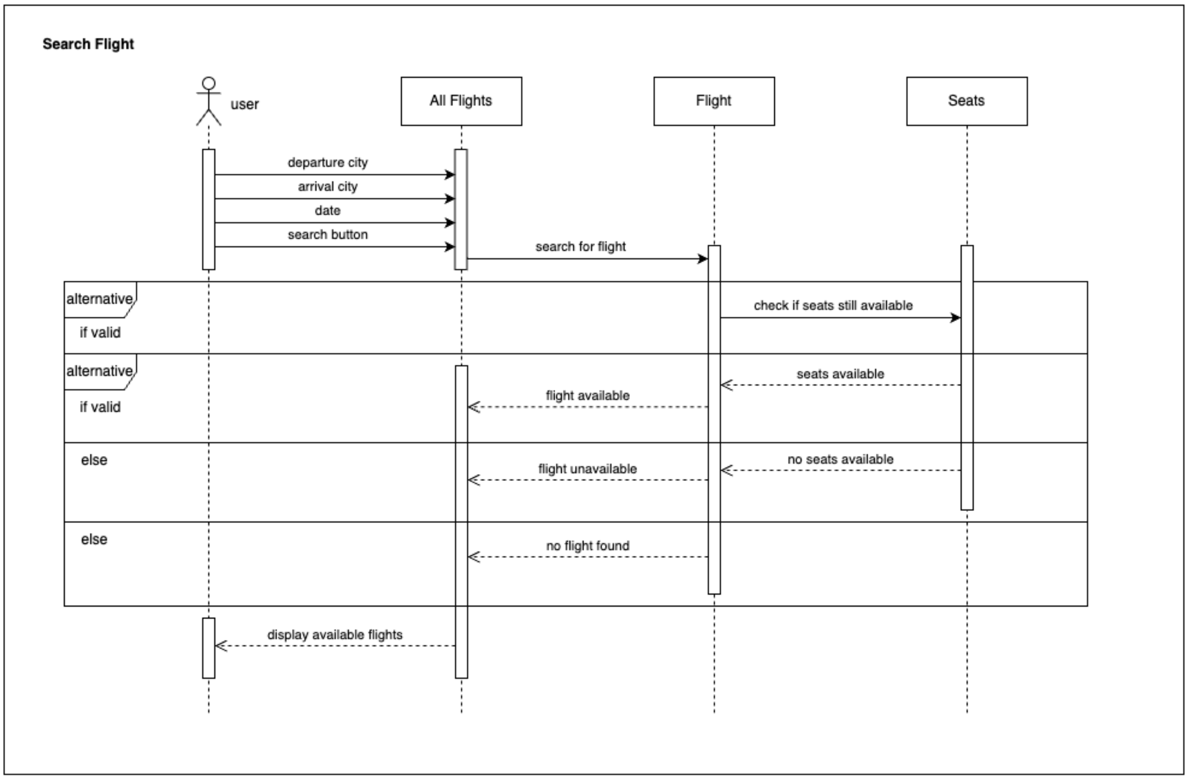
- Not showing every validation case ex) System confirming inputs at each activity.
- Not showing looping decisions for making multiple changes when possible ex) claiming multiple member benefits, or airline agents making multiple modifications.
- It is assumed the user can exit the program at each point, so it was not shown at each decision; only at the end of important scenarios/flows.

B4 - Sequence Diagrams:



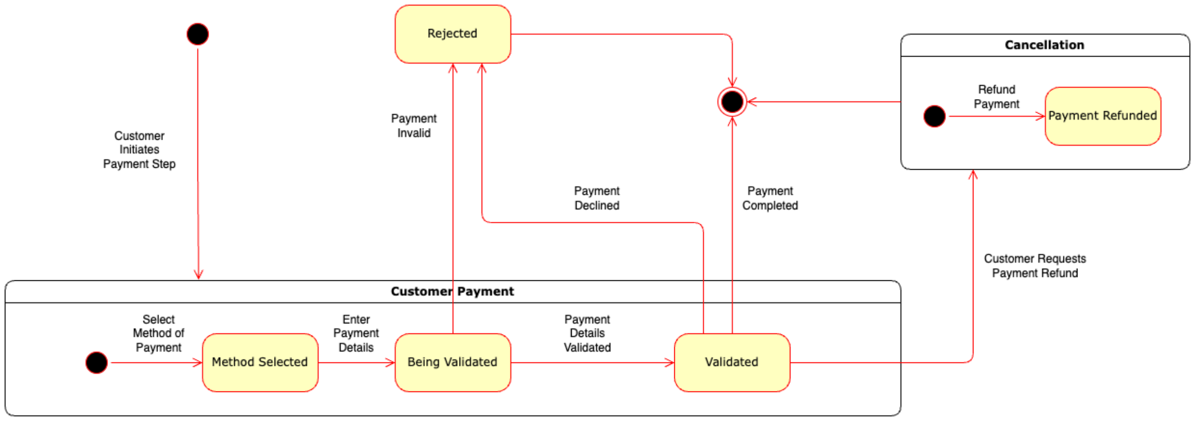



Payment
user
Credit Card
Payment Processor
Payments
enter payment info
validate card
alternative
if valid
card valid
display payment confirmation
mark ticket as paid
else
card valid
display error message, refresh page

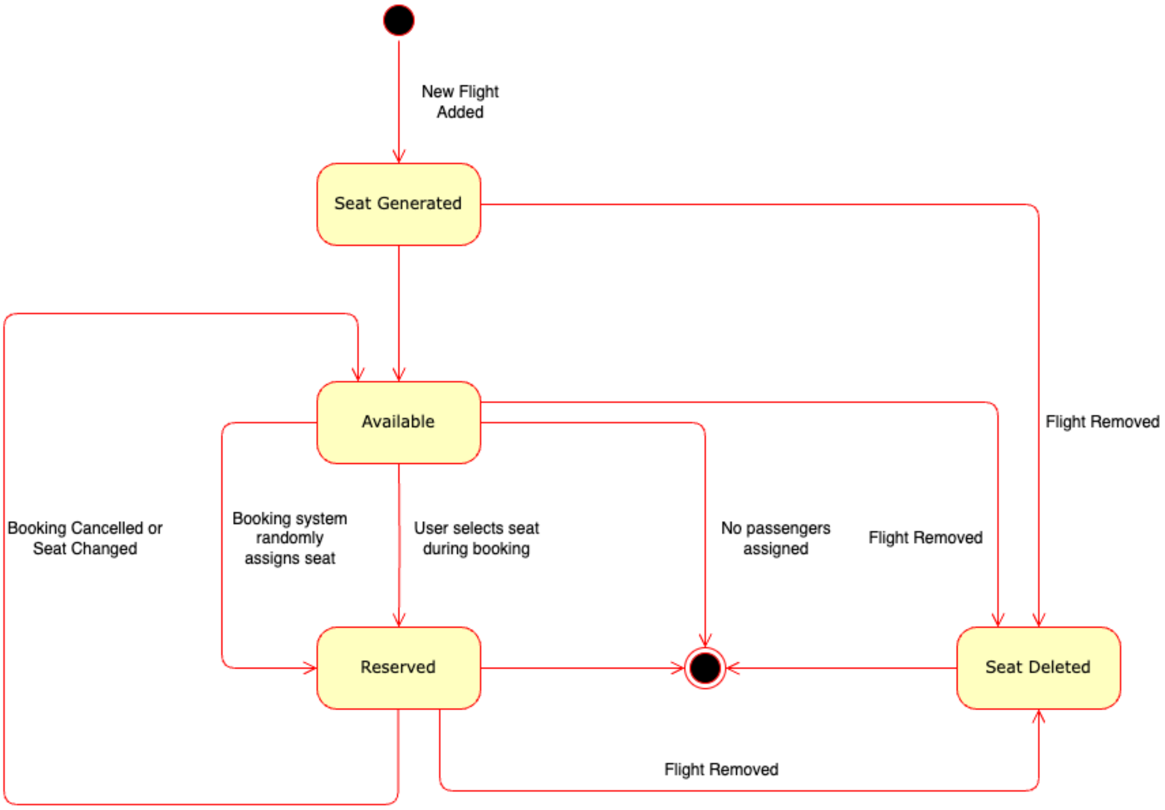

Search Flight
user
All Flights
Flight
Seats
departure city
arrival city
date
search button
search for flight
alternative
if valid
check if seats still available
alternative
if valid
seats available
flight available
else
no seats available
flight unavailable
else
no flight found
display available flights

B5 - State-Transition Diagrams:

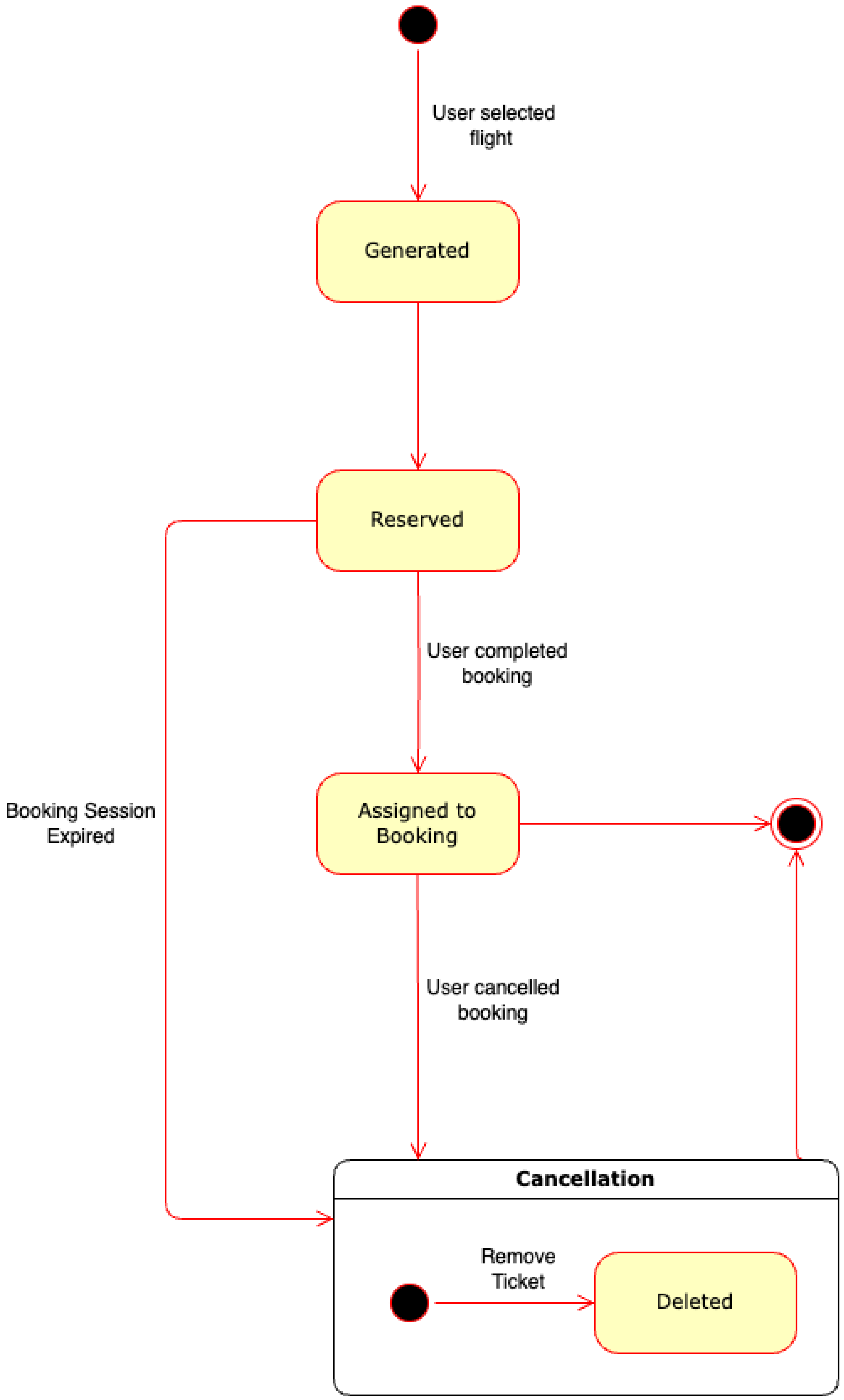
Payment State-Transition Diagram:



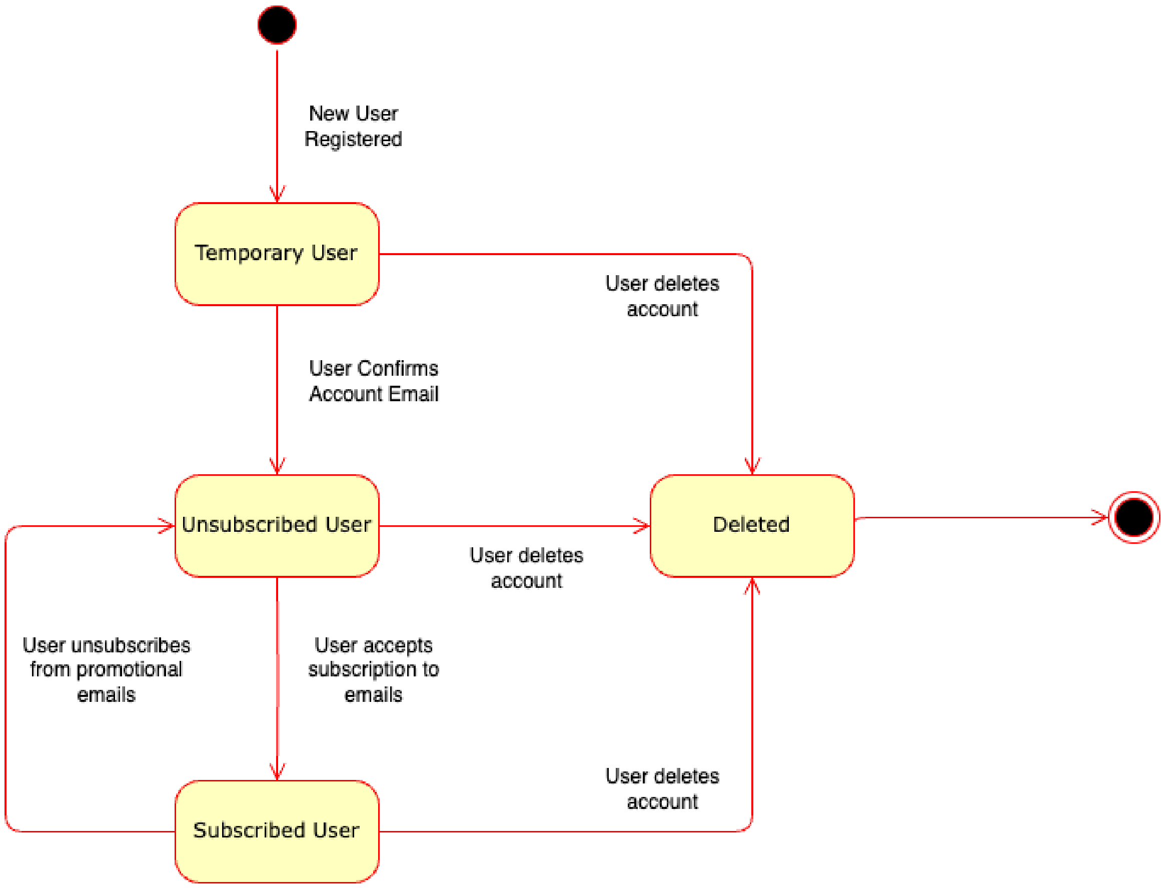
Seat State-Transition Diagram:



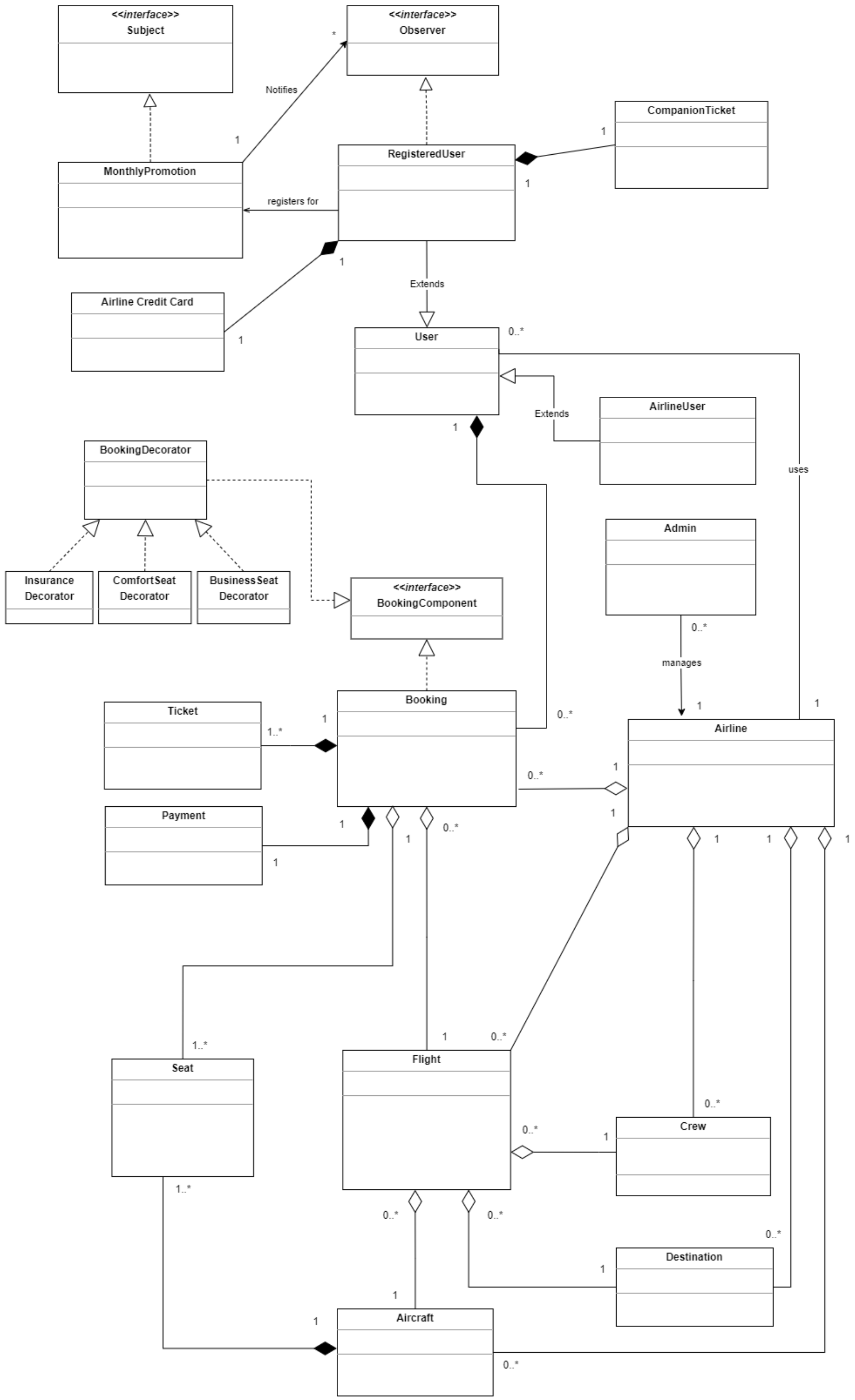
Ticket State-Transition Diagram:



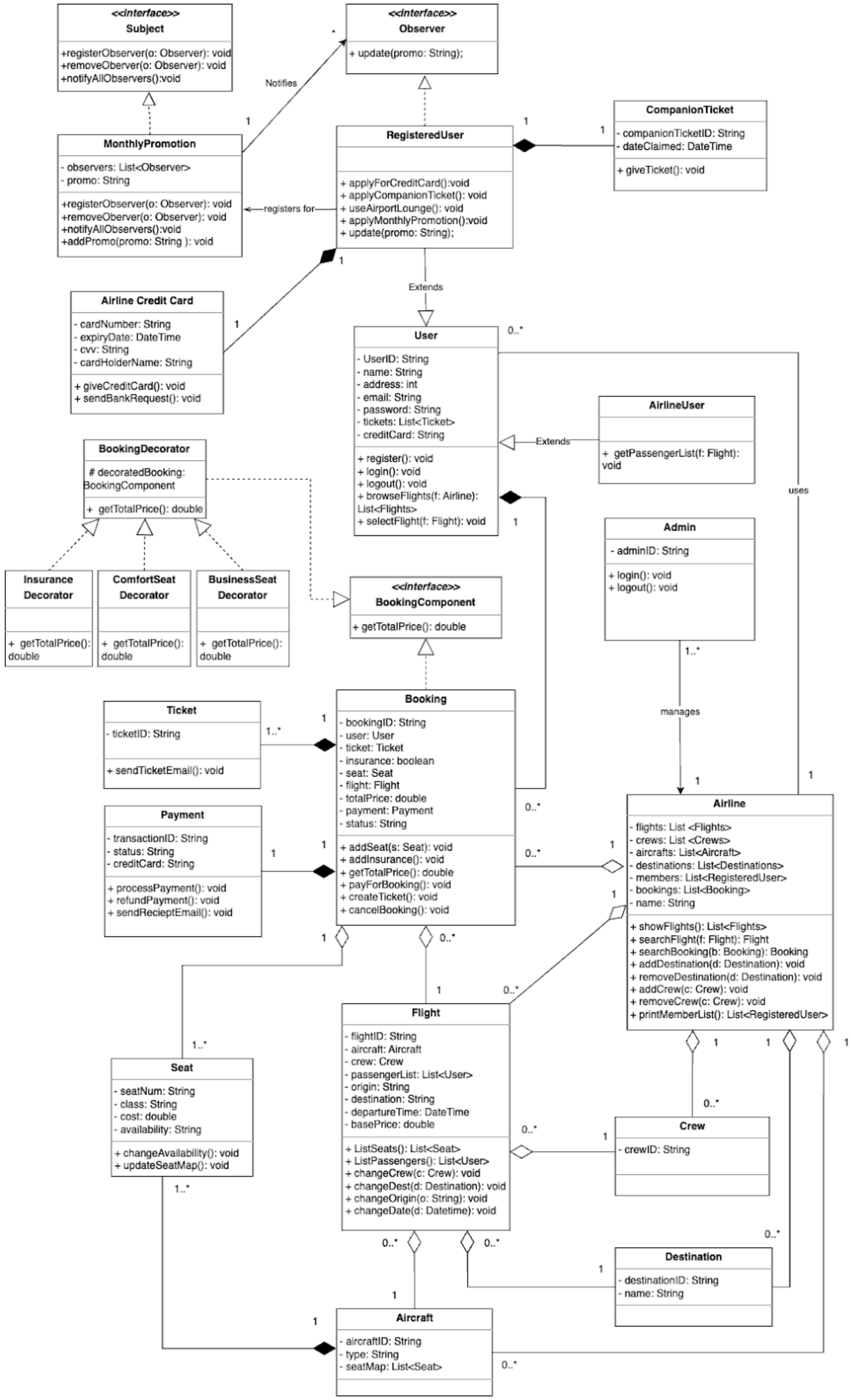
User State-Transition Diagram:



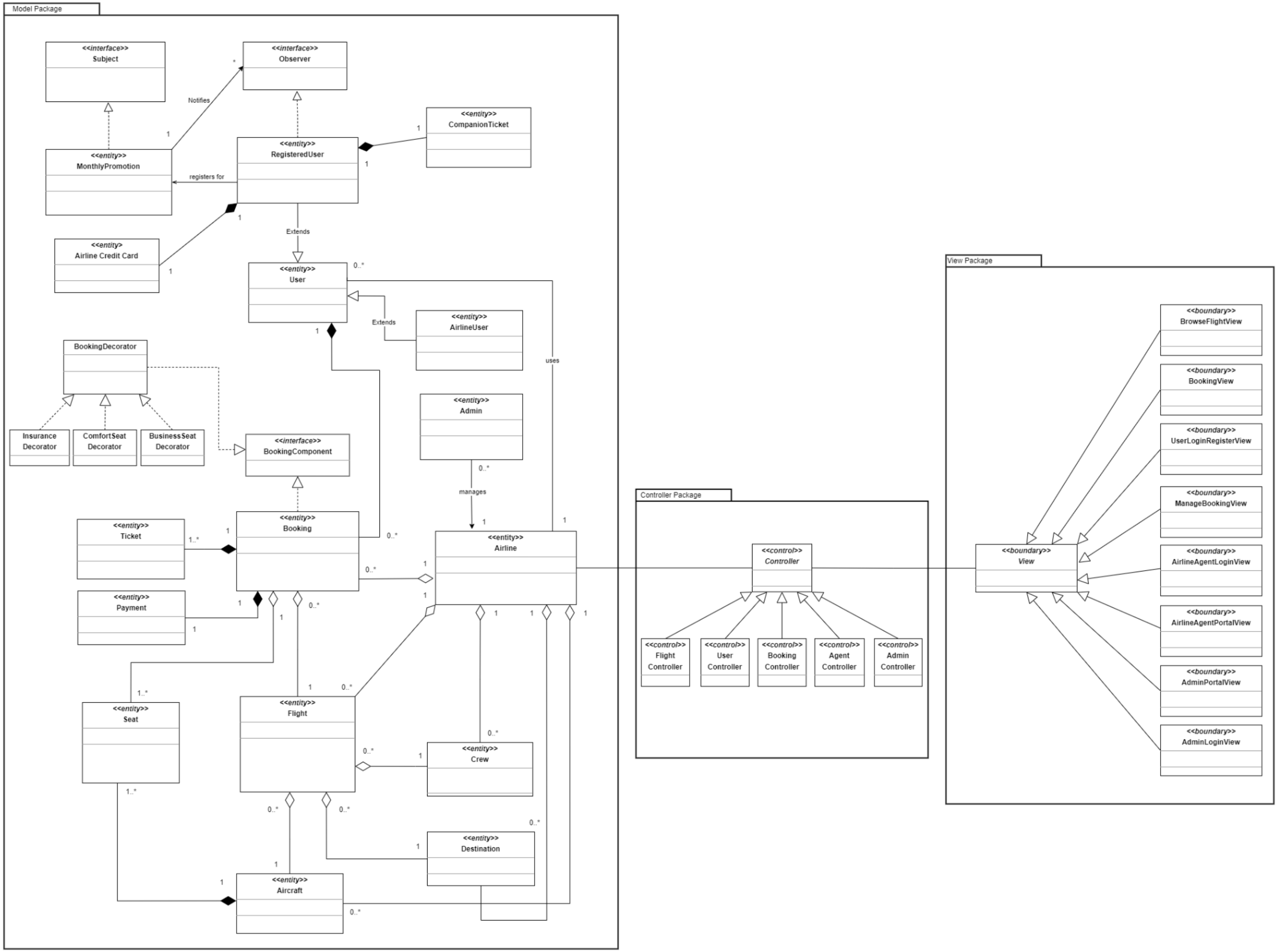
B6 - System’s Domain Class Diagram:



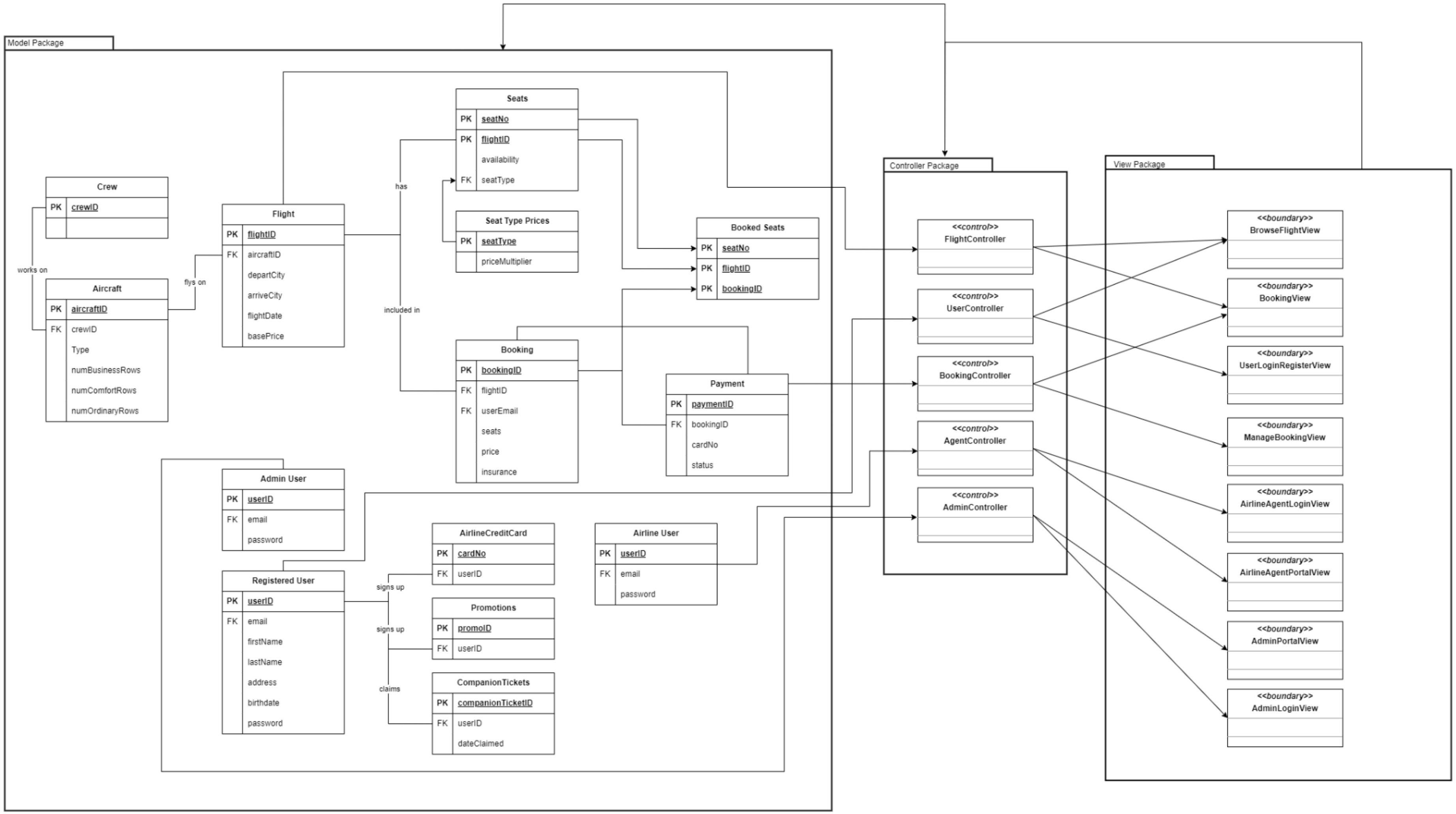
B7 - System’s Domain Class Diagrams (w/ relationship, attributes):



Part C - System’s Detailed Design-Class Diagram:

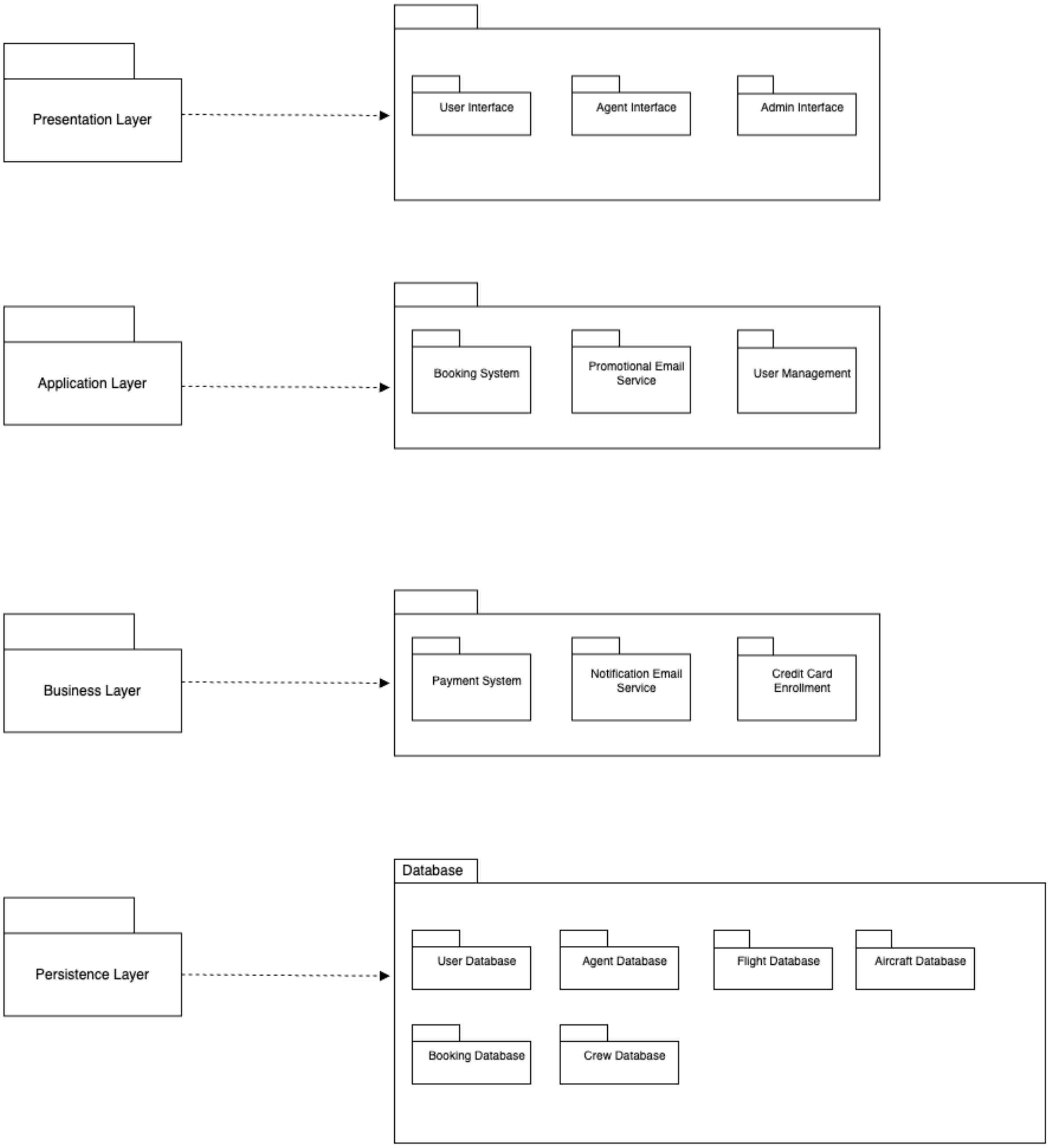


Part C - Design-Class Diagram Implemented (explained in video part 2):

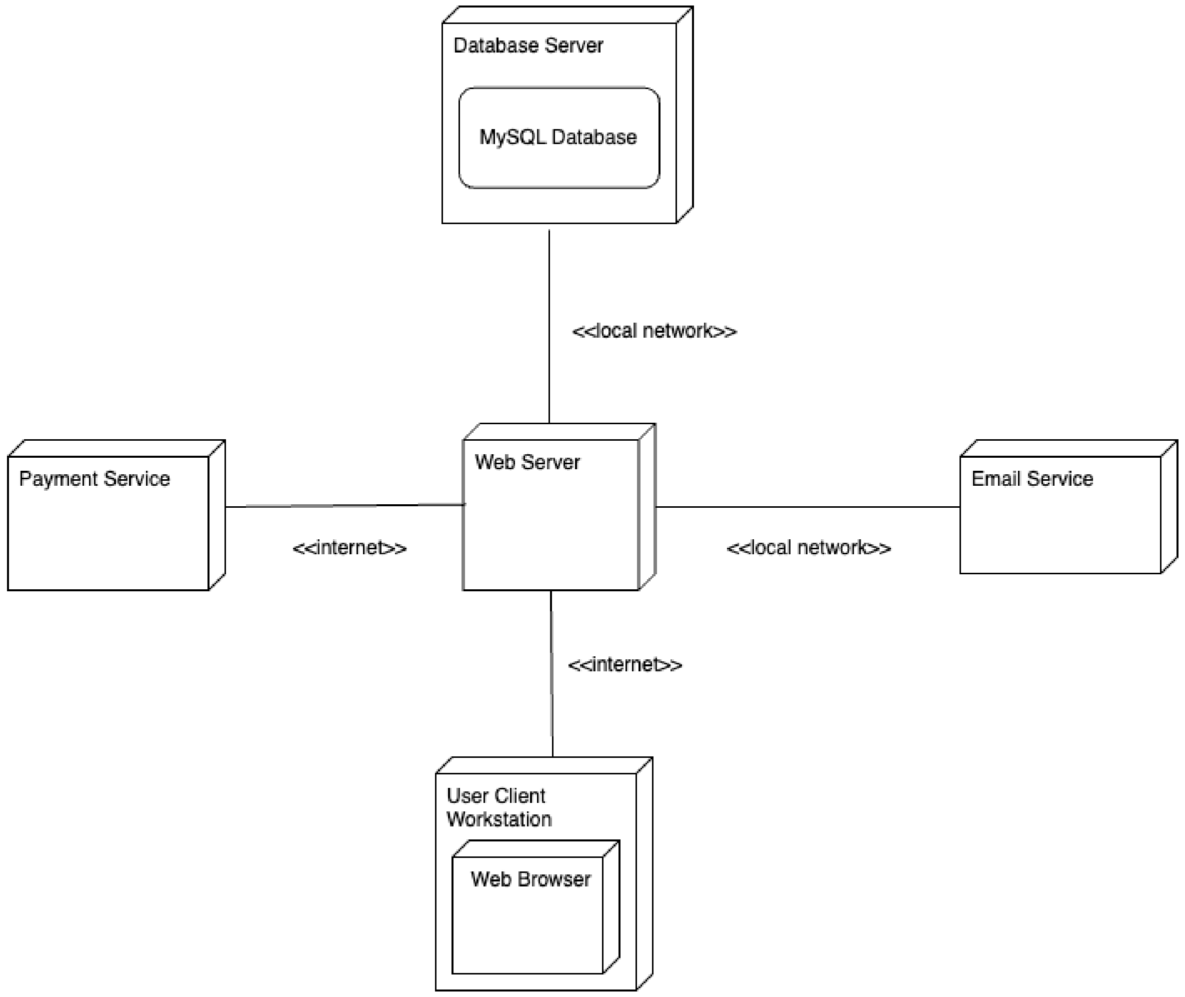


Part D - High-Level System's Architecture:

D1 - Package Diagram:



D2 - Deployment Diagram:


Database Server
MySQL Database
<<local network>>
Payment Service
Web Server
Email Service
<<internet>>
<<local network>>
<<internet>>
User Client Workstation
Web Browser