

ENSF 480 Project

Group 7

Uruba Alam

Alexey Grekov

Kun Lu

Youssef Saad

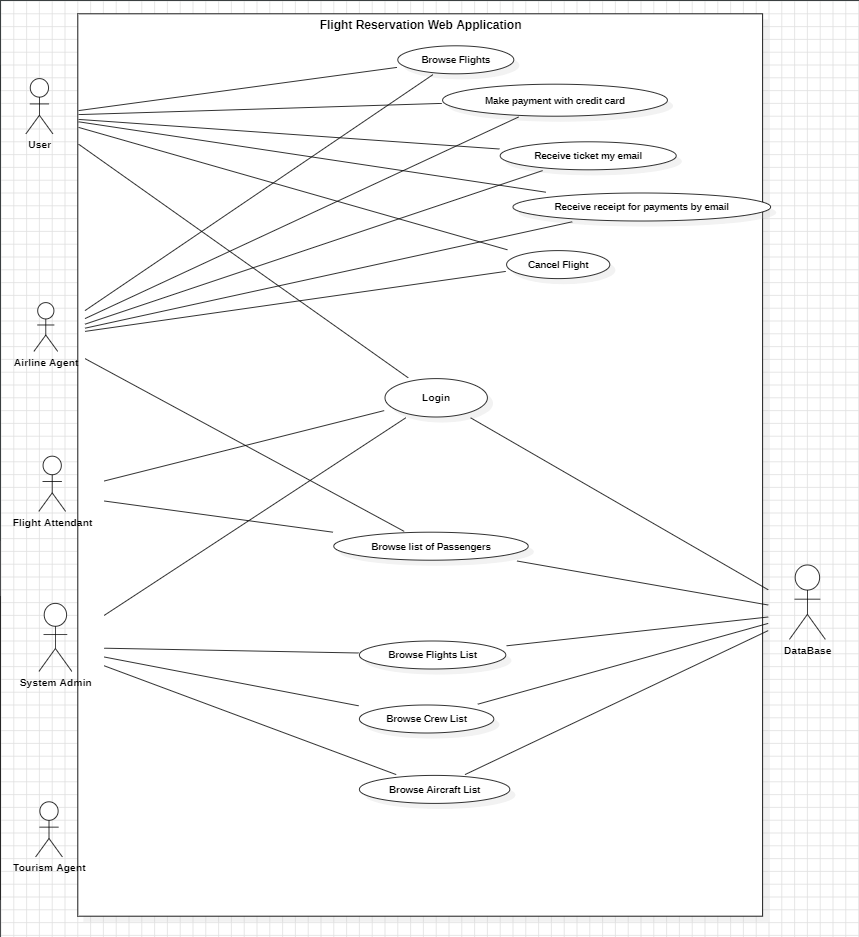
**System’s Description – Youssef**

This Flight Booking System is a comprehensive web-based application for many customers to use to book flights to various destinations. Developed using modern web technologies such as React.js for the frontend and Node.js for the backend, the system is very user-friendly with an interactive interface, accessibility via web browsers, and data storage using MySQL. Its primary purpose is to allow users to search for available flights based on specific criteria such as date, or as mentioned above, destination, then make the corresponding secure booking. Memberships are also available for the wanderlusts! The system comprises several key modules. Firstly, the Flight Search module enabling users to browse available flights with details on schedules, routes, and ticket prices. Secondly, a Booking Management module for users to confirm, modify, or cancel their flight reservations.

Moreover, the system employs Class Diagrams that depict the structure and relationships between different classes or components in the codebase, aiding in software development and maintenance. Use Case Diagrams outline the interactions between different system actors (users, administrators) and the system's functionalities, providing a high-level view of how users interact with the system. Sequence diagrams detail the flow of actions between users and the system, illustrating step-by-step processes for tasks like booking a flight or cancelling a reservation. Deployment diagrams outline the system's architecture, including server configurations and network setups required to host and run the application.

Overall, this airplane booking system aims to provide a seamless and efficient booking experience for users.

**Initial System’s Use Case Diagram – Alexey, Kun, Uruba, Youssef**



**Updated Use-Case Diagram - Youssef**

**A diagram of a flight reservation

Description automatically generated**

**System Scenarios – Youssef**

Browse Flights

When the main page is loaded, the many flight options are available. You can input specific flight options like date or destination

* Customer(actor)
* Flight Search Controller
* Seat options
* Flight List

Payment

After flight is selected, customer will be directed toward payment. Here they can use their saved information if they are a member. Otherwise, they can just pay and then they will in turn receive an email with confirmation and receipt. In the payment process, the user has the option to pay an additional fee for ticket insurance. Afterwards, the database is updated with this information so that the respective seats are occupied and prevent the chance of double booking.

* Customer(actor)
* Flight
* Seat
* Payment

Login/Register

A button to log in or register is available. This will give users access to their respective privilege level. After logging in, they can proceed with whatever action is logically next. If a client doesn't want to register for an account, they can continue as a guest. A credit card option is available. If the user registers an account they can get monthly news, free yearly tickets and discounted lounge prices. Personal details are required for register as they are stored for future use. Login requires username and password to ensure security.

* Any user (actor)
* Login
* Register
* Membership

Browse Passenger List

A privileged view of all the passengers on a flight. This allows printing a list to confirm each passenger is seated correctly and to resolve disputes. The list displays each passenger’s personal and flight information associated with each flight. Each flight has a unique list. If any information needs to be edited/removed/added, the database will be updated immediately after and only admin can make changes.

* Flight Attendant or Admin (actors)
* Flight
* Passenger list

Cancel Flight

An option to cancel flight if the need arises is available. This is doable by first logging in to view previous orders. From here, the actor can click on delete flight, and after confirming the deletion, the database is updated so that the flight is available to be booked by others.

* Customer(actor)
* Cancel Flight
* Previous orders
* Login

Browse flight list, Browse crew list, Browse Aircraft list

Privileged access allows all the flight/crew/aircraft information to be changed. From here, the admin can edit/delete/add a flight/crew/aircraft based on whatever conditions are in the outside world. Only the admin can do this, which needs login/verification, and can prevent issues from having users who shouldn't have access be able to change things. All edits are updated directly in the database so they are viewed by the system users.

* Admin(actor)
* Edit/delete/add 🡪 flight/crew/aircraft
* Login
* Flight/crew/aircraft list
* Destinations
* Date

**System’s Conceptual Model – Kun and Youssef**

**Domain Class Diagram - Youssef**

* Use Observer Pattern to notify DB of changes made (add/remove/update)
* Use Strategy Pattern for all the different browsing. There can be one main browsing layout and use strategy to include different information based on use case
* Use Singleton Pattern for login information
* Use Decorator Pattern for GUI

**System Domain Diagram with Relations**

**A diagram of a network

Description automatically generated**

### 

**System Architecture - Youssef**

**Activity Diagram – Alexey and Youssef**

**A diagram of a diagram

Description automatically generated**

**Sequence Diagrams - Youssef**

**State Transition Diagrams - Youssef**

**Payment**

**A diagram of a program

Description automatically generated**

**Browse Flights**

**A diagram of a program

Description automatically generated**

**Login**

**A diagram of a system

Description automatically generated**

**Browse Passenger List**

**A diagram of a flight

Description automatically generated**

**Cancel Flight**

**A diagram of a process

Description automatically generated**

### 

### 

**Deployment Diagram - Youssef**

**A diagram of a server

Description automatically generated**

**Package Diagram - Youssef**