Software Requirement and Design Specifications

[Airline Reservation System]:

Version: [1.001]:

Course Code:
(SE 2002)

Instructor:
Miss Syeda Rubab

Project Team:

Arqam Hasan
Suleman Mohiuddin
Talha Fuzail

Submission Date:
13th Mar 2023

1. Introduction

1.1. Purpose of Document

This SRS document presents a detailed description of the ARS. It represents the client requirements analysis that defines the functional and non-functional requirements of the airline website and its different functionalities. Its purpose is to define the requirements involved in developing the system.

1.2. Intended Audience

Any individual above the age of 18 who wishes to search, book or reserve flights to any destination or to check availability of Airlines.

1.3 Definition of Terms, Acronyms and Abbreviations

- ARS: Airline Reservation System

1.4 Document Convention

- Main Headings Font Size is 25 Calibri (Bold & Italic).`
- Subheadings Font Size is 15 Calibri (Italic).
- General Font Size is 12 Calibri (Body).
- All Headings and Subheadings have Hierarchical Numbering.

2. Overall System Description

2.1. Project Background

For companies and/or airports looking to digitize their flight management system and booking processes to minimize human error, and to utilize that need in the market to simultaneously make profit.

2.2. Project Scope

The ARS provides a verified user with the generic reservation options to book flights.
The ARS will perform these tasks: -

- The ARS can be used to check for availability of the flight tickets for the specified flight, destination and date and time of journey.
- The ARS will provide the facility to book user specified tickets if available.
- The ARS will send verification emails and confirmation emails on booking.
- The ARS provides the function to reschedule flights.
- The ARS provides a cancel module to users in case they want to cancel flights.

2.3. Not In Scope

The functionalities that aren't included in the ARS scope include: -

- The ARS will not allow refunds of flights.
- ARS is not affiliated with any specific airline.
- The ARS does not provide on flight meal information or booking
- The ARS does not work without a stable internet connection.

2.4. Project Objectives

The objectives of the ARS are to: -

- Minimize human error in management and reservation of flights.
- Minimize tickets loss/frauds

- Provide a more reliable and efficient software to Airports and Travel Companies.
- Utilize the need for a better digital system and make profit

2.5. Stakeholders

The Stakeholders include: -

- Users
- Airport Management
- Software Development Team
- Marketing Team
- Server Providers
- Bank
- Sales Team
- Government
- Q/A Testers
- Travel Agents

2.6. Operating Environment

The ARS will operate in all windows operating systems, Windows 7 and above with DOS support. The system must have at least 5 GBs of free space and a stable internet connection.

2.7. System Constraints

The ARS comprises of a few imposed constraints which include: -

- The ARS operates only in Windows
- The ARS has only English Language support.
- The ARS only accepts pay online through PayPal.
- The ARS only functions with a stable internet connection.
- The ARS UI is developed for ages 16 and above.

2.8. Assumptions & Dependencies

- It is assumed that the user is using a system above Windows 7.
- It is assumed that the user has a stable internet connection.
- The user has general knowledge of working with UI software.
- The system should run 24/7.

3. External Interface Requirement

3.1. Hardware Interface

The program is to be used on a windows computer with java lib installed and operated through a mouse. The person must click on the login button after inserting the correct login credentials, if not registered the person must input a long integer that is their passport number to register. afterwards only a mouse will be used to further navigate through the application.

3.2. Software Interfaces

- Mysql

used for storing all the registered users info and also all the flight details and seats booked

- Java JDK/SDK

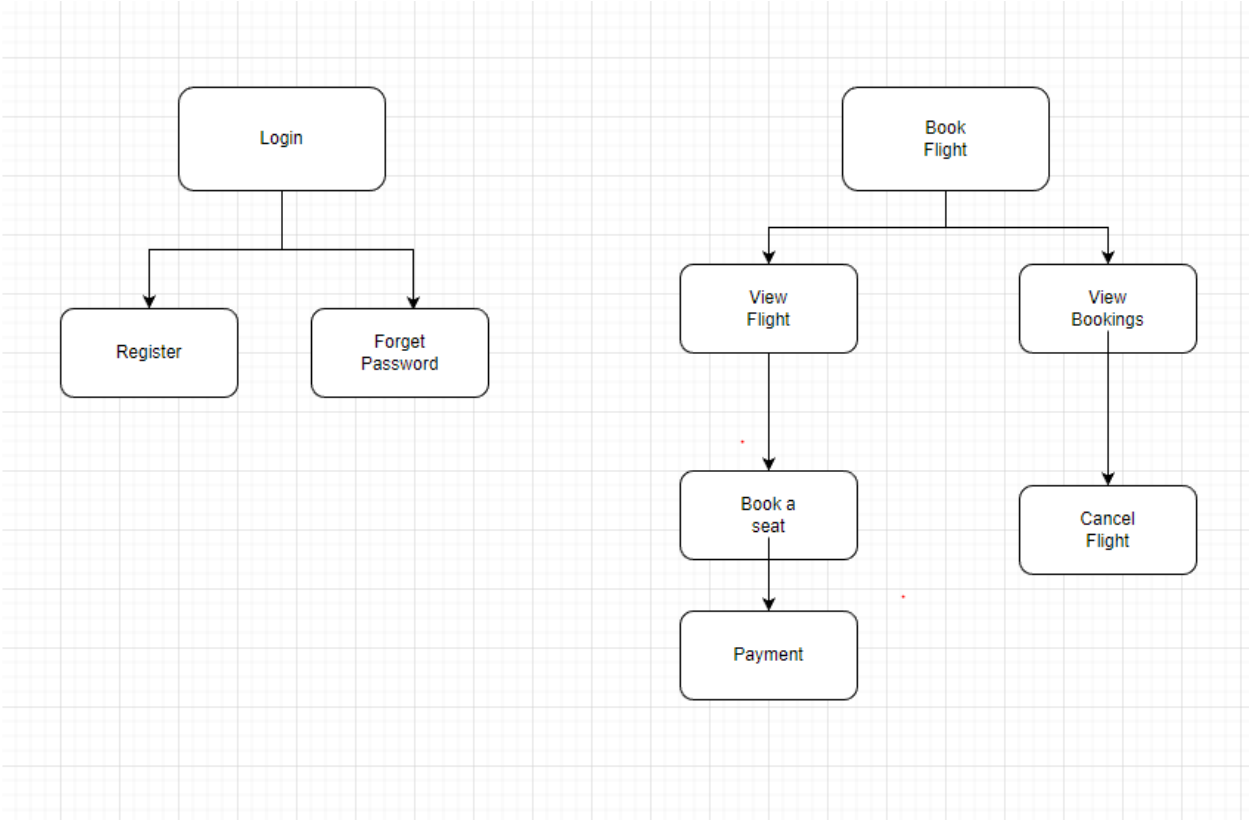
Used to run the application as the program will be written in Java.

3.3. Communications Interfaces

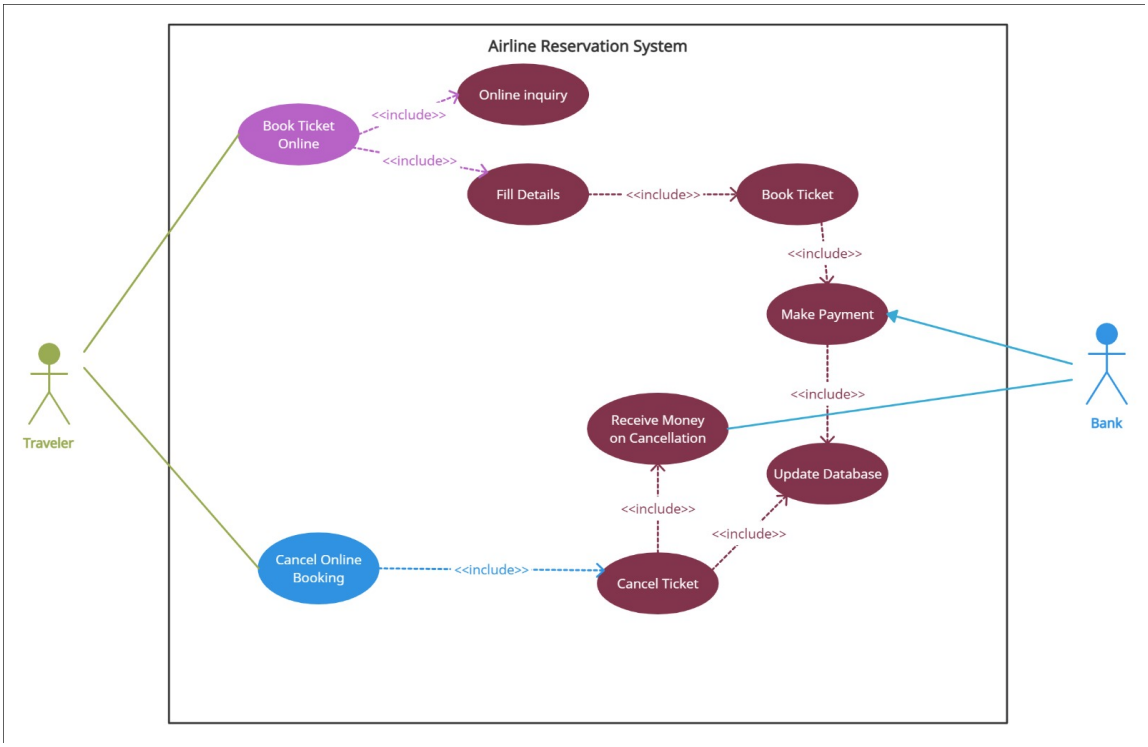
- Internet communication will be used to connect to the database servers and keep the app updated
- Email communication will also be used as the ticket will be sent via emails.

4. Functional Requirements

4.1. Functional Hierarchy



4.2. Use Cases Diagram



4.2.1. Use Cases Description

Use case Name: Airline reservation system

Use case description:

A system that allows users to Book flights online and keep track of Booked flights easily meanwhile letting the Airport Administration have control over flight schedules and seats availability.

Primary actors:

- Customers

- Admin

Secondary actors:

- Bank
- Airline Servers

StakeHolders:

- Users
- Airport management
- Software development team
- Marketing team
- Server providers
- Bank
- Sales team
- Government
- Q/A testers
- Travel agents

RelationShips:

- Includes:
- Ticket booking / canceling
- Choose a seat of choice.
- Select payment option
- Extends:
- View flights.
- View prices
- Acquire information about flights.

Pre-Conditions:

- Must be registered with a valid passport number to book flights.
- Must have internet connection.
- Must not be a defaulter.
- Must be using a compatible device.

Flow of Events:

- Customer logs in and if he/she is not a defaulter they are allowed access.

- If not registered the person must get registered by using a valid passport number.
- Once logged in the user can select any flight that is scheduled and can even select the seat of their choice if it is available.
- Once the desired flight and seat is booked the customer is asked to pay via credit/debit card to confirm the booking.
- After all the process an E-ticket is issued to the customer.
- Booking can only be canceled 2 days prior to the flight.

Alternative and Exceptional Flow:

- Admin logs in.
- Can upload new flight schedules / edit previous ones/ delete a

flight schedule.

- Can mark seats as booked or not.
- View customers' information.

Post-Conditions:

- Must have an appropriate amount in the bank.
- Must show the ticket to board a plane.
- Can call customer service.

4.2.1.a. Use Cases Description (Canceling ticket)

UseCaseID:		Cancel Reservation
Goal In Context:		A customer wishes to cancel a reservation.
Scope:		Airline Reservation System (ARS)
Pre-Condition:		A reservation has already been made. Actor has successfully navigated to the main options screen.
Success End Condition:		The selected reservation has been cancelled.
Failed End Condition:		The selected reservation has **NOT** been cancelled.
Primary Actor:		Customer
Trigger Event:		Selects the "Cancel Reservation" option.
Main Success Scenario		
Step	**Actor**	**Action Description**
1	Customer	Selects the "Cancel Reservation" option.
2	ARS	Displays a screen with an input field for a reservation number.
3	Customer	Enters a reservation number and clicks the "Submit" button.
4	ARS	Displays the details of the reservation.
5	Customer	Selects the "Process Cancellation" option.
6	ARS	Displays the main options screen. The message "Reservation Successfully Cancelled" is also displayed.

Scenario Extensions

Step	**Actor**	**Action Description**
2.a	Customer	Selects the "Cancel" option.
2.a.1	ARS	Displays the main options screen.
3.a	Customer	Selects the "Cancel" option.
3.a.1	ARS	Displays the main options screen.
5.a	Customer	Selects the "Abort" option.
5.a.1	ARS	Displays the main options screen.

Scenario Variations

Step	**Actor**	**Action Description**
None		

4.2.1.b. Use Cases Description (Reservation)

UseCaseID:	Make Reservation
Goal In Context:	A customer wishes to make a reservation.
Scope:	Airline Reservation System (ARS)
Pre-Condition:	Actor has successfully generated a list of flights as documented in the "Search for Flights" use case.
Success End Condition:	A reservation has been made.
Failed End Condition:	A reservation has **NOT** been made.
Primary Actor:	Customer
Trigger Event:	Selects the "Book This Flight" link from the list of flights presented.

Main Success Scenario

Step	**Actor**	**Action Description**
1	Customer	Selects the "Book This Flight" link from the list of flights presented.
2	ARS	Displays a screen with input fields for: a credit card number, cardholder name, and credit card expiration date.
3	Customer	Enters a credit card number, the cardholder name and the credit card expiration date and clicks the "Submit" button.
4	ARS	Displays the details of the reservation.
5	Customer	Selects the "Complete Reservation" option.
6	ARS	Displays the main options screen. The message "Reservation Successfully Made" is also displayed.

Scenario Extensions		
Step	**Actor**	**Action Description**
3.a	Customer	Does not complete all the necessary fields.
3.a.1	ARS	Redisplays the input fields from step 2 with the input data and a note stating that certain required data is missing or invalid.
3.b	Customer	Selects the "Cancel" option.
3.b.1	ARS	Displays the main options screen.
5.a	Customer	Selects the "Cancel" option.
5.a.1	ARS	Displays the main options screen.
Scenario Variations		
Step	**Actor**	**Action Description**
None		

4.2.1.c. Use Cases Description (Search for Flights)

UseCaseID:	Search for Flights
Goal In Context:	A customer wishes to search for available flights between two cities for the chosen dates.
Scope:	Airline Reservation System (ARS)
Pre-Condition:	Actor has successfully navigated to the main options screen.
Success End Condition:	A list of flight matching the search criteria is presented.
Failed End Condition:	A list of flight matching the search criteria is **NOT** presented.
Primary Actor:	Customer
Trigger Event:	Selects the "Search Flights" option.

Main Success Scenario

Step	**Actor**	**Action Description**
1	Customer	Selects the "Search Flights" option.
2	ARS	Displays a screen with input fields for source airport, destination airport, : type of ticket (one way or round trip) and the departure and arrival date based on the choice of ticket type. Two-way ticket will force an arrival date to be entered.
3	Customer	Enters the all information and clicks the "Search" button.
4	ARS	Displays a listing of all flights that match the customers input from step 3, each item in the list has an associated price and a "Book This Flight" link.

Scenario Extensions		
Step	**Actor**	**Action Description**
3.a	Customer	Selects the "Cancel" option.
3.a.1	ARS	Displays the main options screen.
3.b	Customer	Does not complete all the necessary fields.
3.b.1	ARS	Redisplays the input fields from step 2 with the input data and a note stating that certain required data is missing.
Scenario Variations		
Step	**Actor**	**Action Description**
None		

5. Non-functional Requirements

5.1. Performance Requirements

- The system will have the capacity to support at least 100 concurrent users.

- The system shall send out verification request immediately (within 100ms) after it receives a user submitted form.

- The system shall update database user information immediately after a new user registers, it will update all flight information whenever new flights are booked.

- Flight verification emails will be sent to the user by the system as soon as the user confirms a booking (within 100ms).

5.2. Safety Requirements

- There will be a secondary backup database in case of any data corruption or loss in the primary database.

- They system will have necessary firewalls to prevent third party intervention.

- Any system failure will be logged, and the backup database will regenerate all data and operations into the primary database.

- All users will have to verify their identity before being able to use the system.

5.3. Security Requirements

- The system urges the user to set a unique password with a minimum of 8 characters and at least one digit.

- Email addresses should be verified before the system grants user access. This verification shall be exercised by sending the prospective user a confirmation email after user registers.

- All exchanges from client to server involving private data shall occur using the highest available level of secure connection (e.g., https).

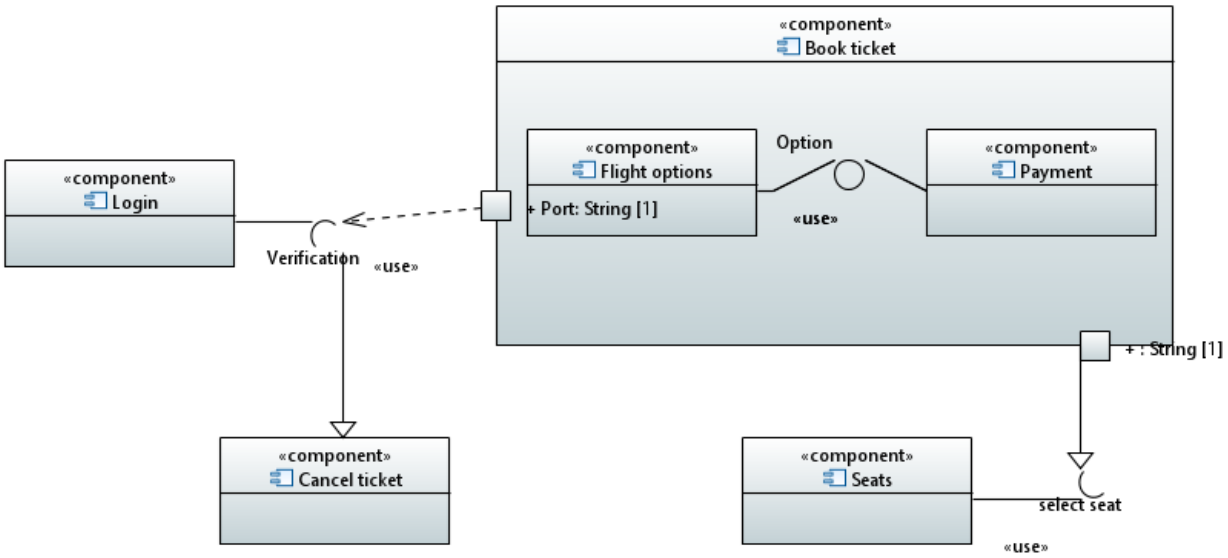
- A user will be logged out after 30 mins of inactivity.

- All databases are encrypted with the highest level of security to prevent third party involvement and/or scams.

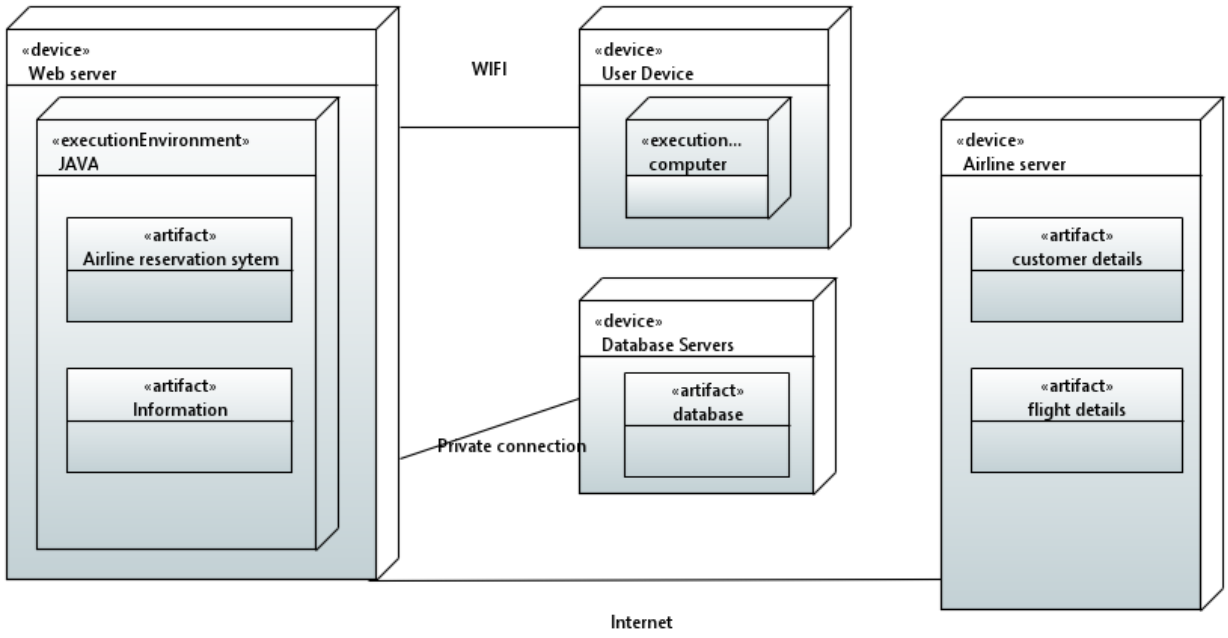
5.4. User Documentation
Along with the software, a user manual will be given and/or online instructions.

6. System Architecture

6.1. System level Architecture



6.2. Software Architecture



7. Design Strategies

The design strategies and decisions that impact the overall organization of the airline reservations system are critical for ensuring the system's scalability, maintainability, and extensibility. Some of these strategies and decisions are:

-Future System Extension or Enhancement: The system should be designed with modularity and flexibility in mind to allow for future system extensions or enhancements. The use of modular components with well-defined interfaces and a clear separation of concerns can help simplify the addition of new features or subsystems in the future.

-System Reuse: The system should be designed to facilitate code reuse and avoid code duplication. This can be achieved through the use of common libraries, frameworks, and design patterns.

-User Interface Paradigms: The system's user interface should be intuitive, easy to use, and consistent with established user interface paradigms. This can help reduce the learning curve for users and improve user adoption.

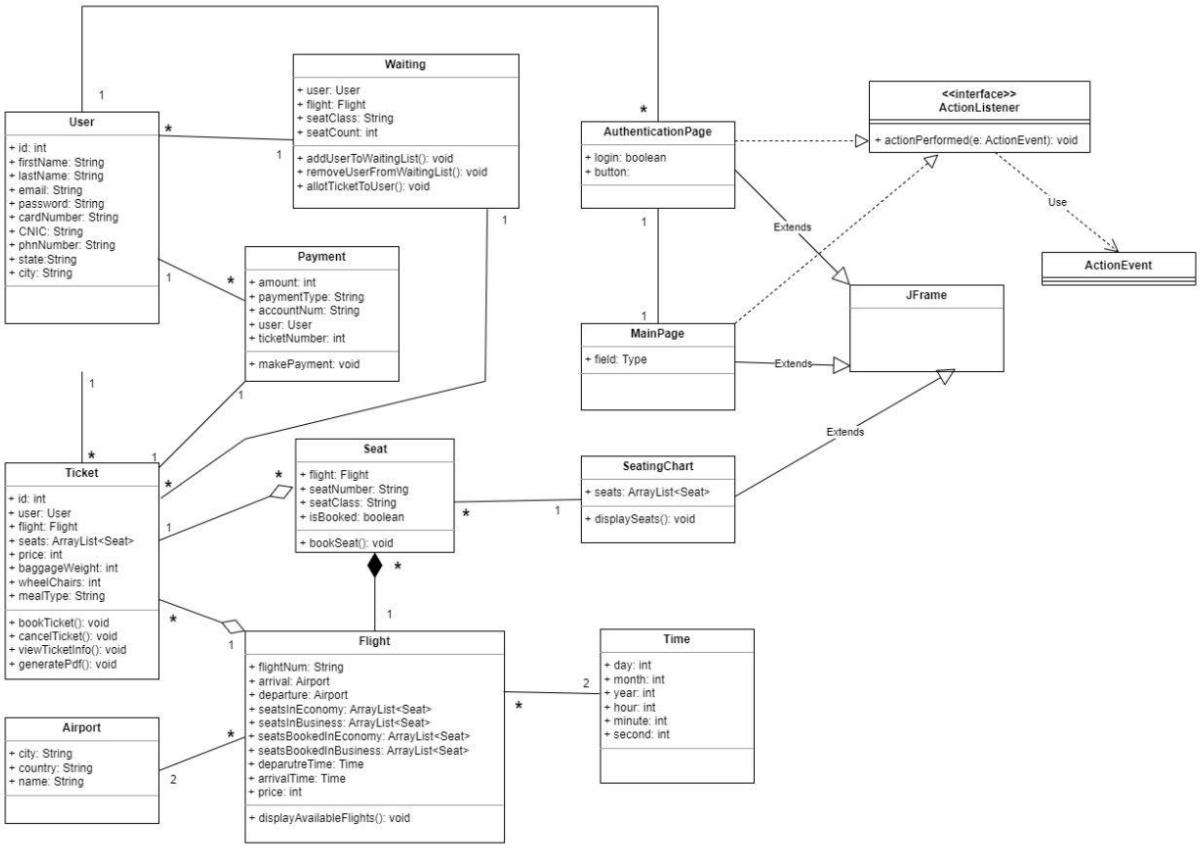
-Data Management: The system's data management strategy should consider issues such as data storage, distribution, and persistence. Data should be stored in a way that enables efficient retrieval and modification, while ensuring data consistency and integrity. Additionally, the system should be designed to accommodate potential data growth and scalability needs.

-Concurrency and Synchronization: The system should be designed to handle concurrency and synchronization issues that may arise due to concurrent access to shared resources. This may involve the use of concurrency control mechanisms such as locking, thread synchronization, or transaction management.

Trade-offs may arise when making these design decisions. For example, prioritizing code reuse and modularity may add additional complexity to the system, which could increase development time and maintenance costs. Similarly, accommodating potential data growth may require a more robust and scalable data storage solution, which could increase storage costs. The design decisions should be based on the system's specific requirements, design goals, and principles, while balancing trade-offs between various design factors.

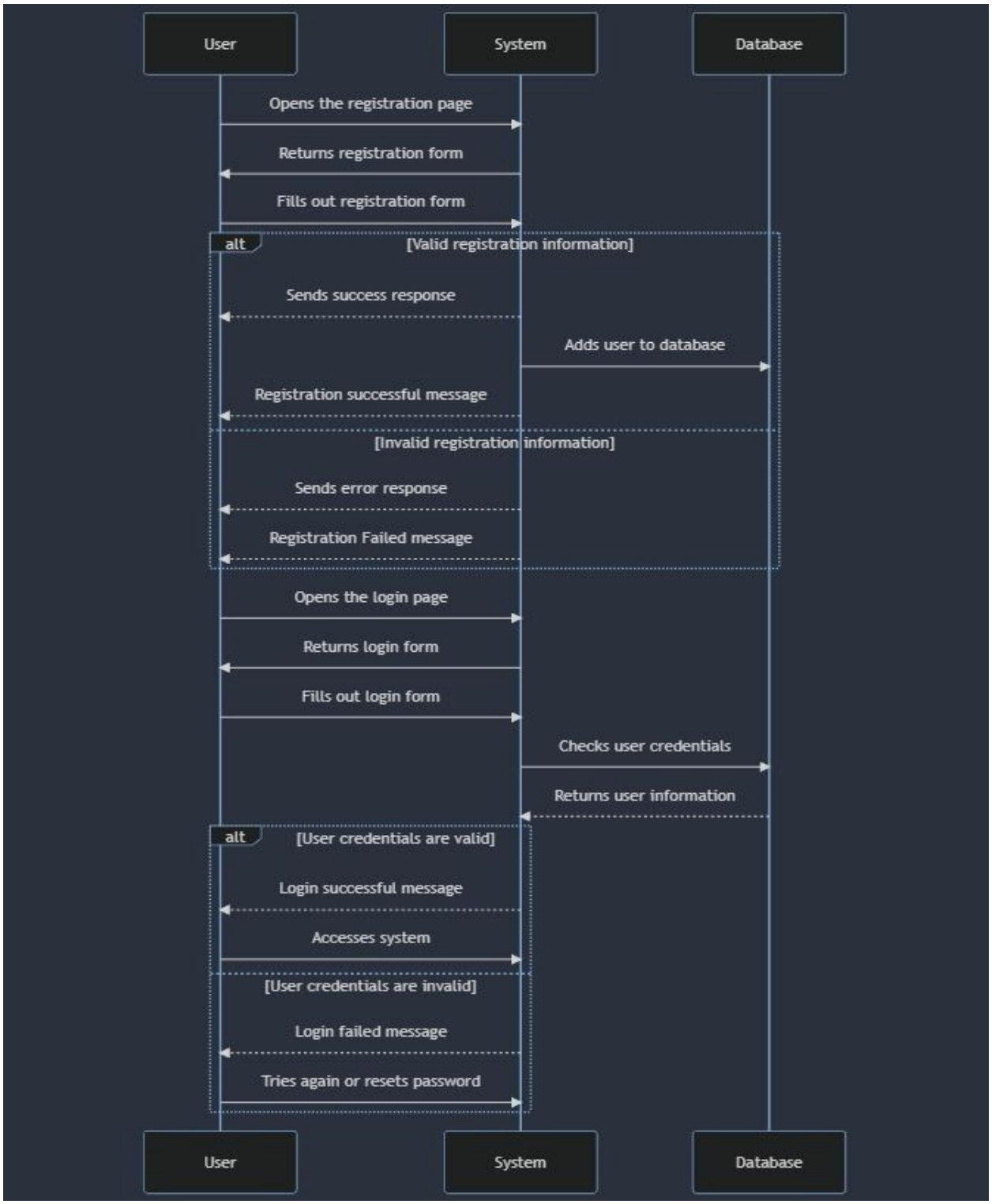
8. Detailed System Design

8.1. Class Diagram



9. Application Design

9.1.1.a Sequence Diagram 1

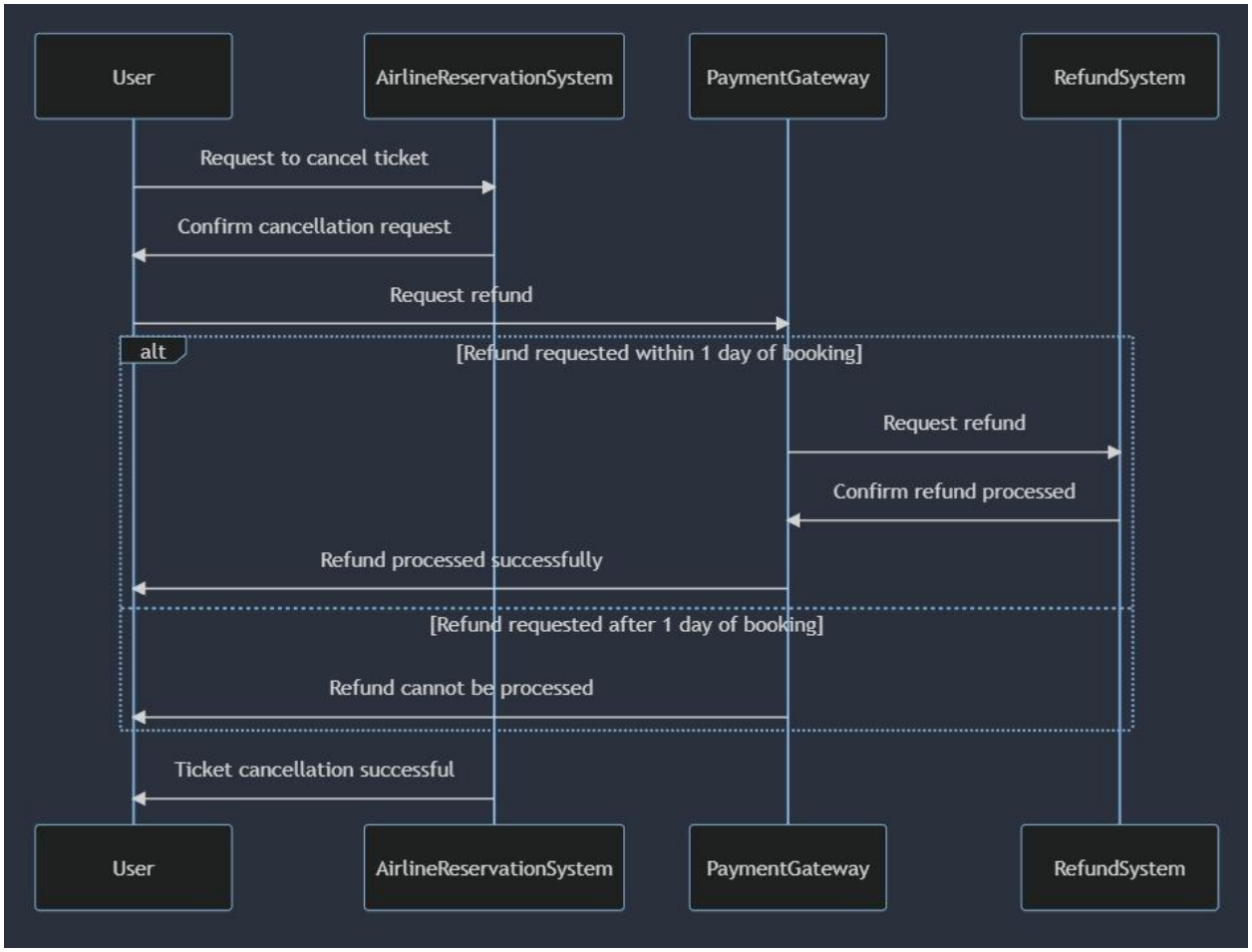


This Sequence diagram represents the login or registration process of our application according to the user.

9.1.1.b Sequence Diagram 2

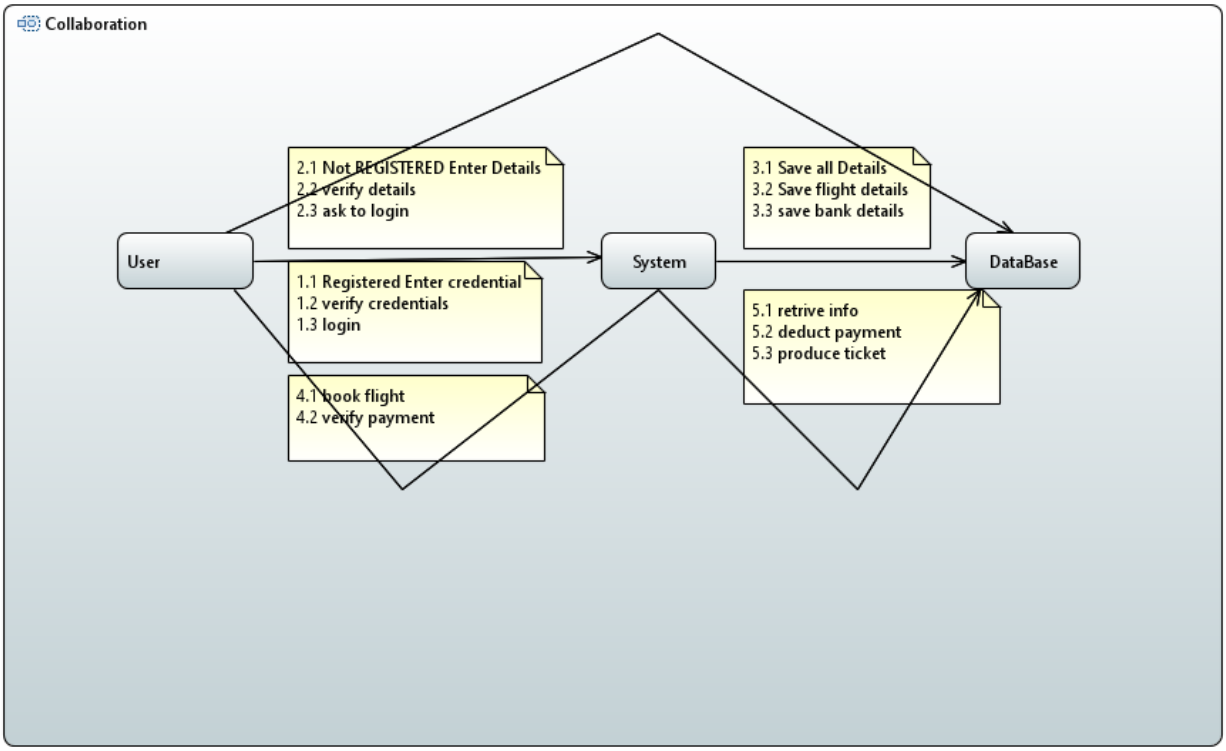
Flight booking

9.1.1.c Sequence Diagram 3



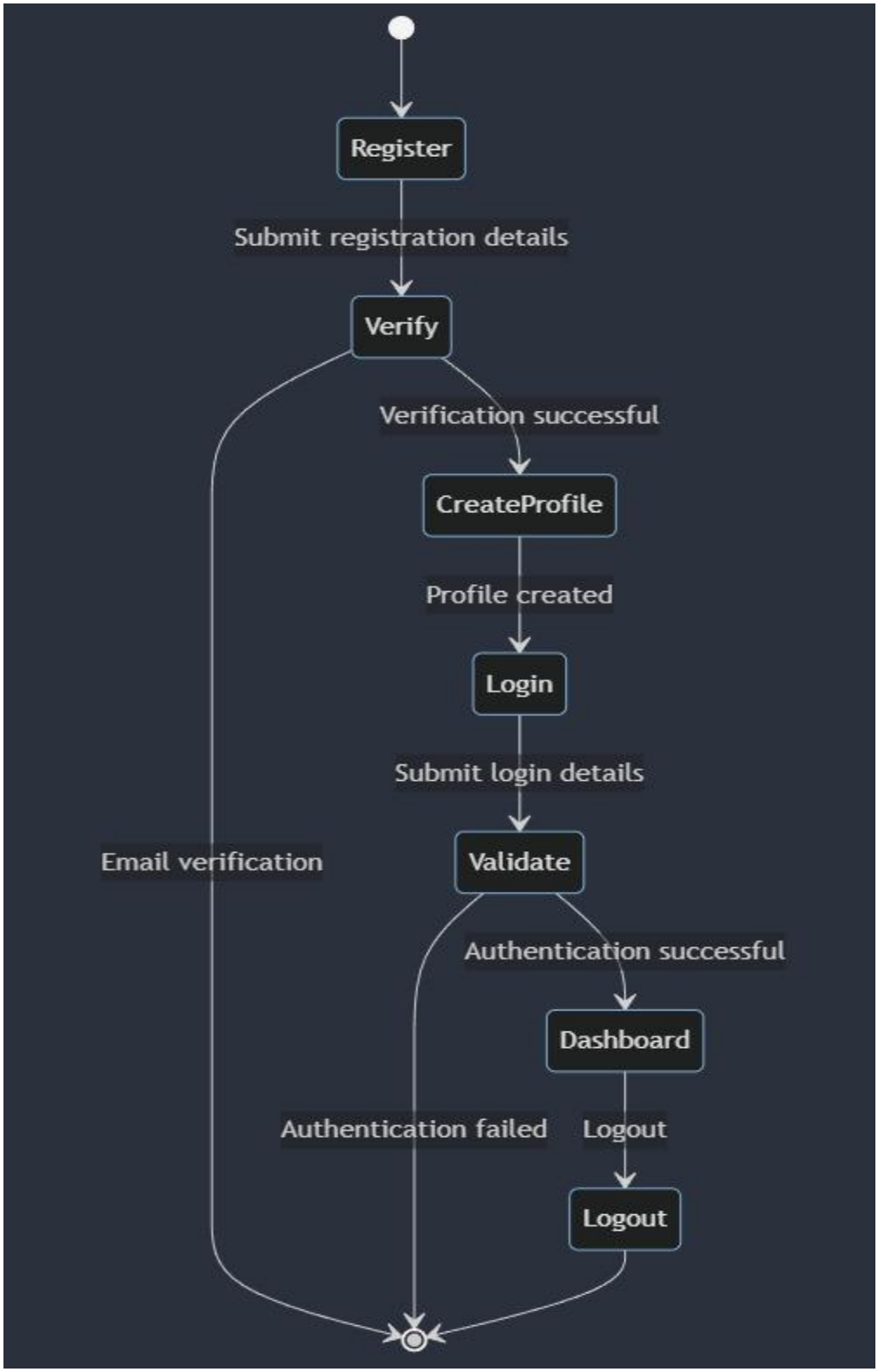
Cancel ticket

9.1.2 Collaboration Diagram.

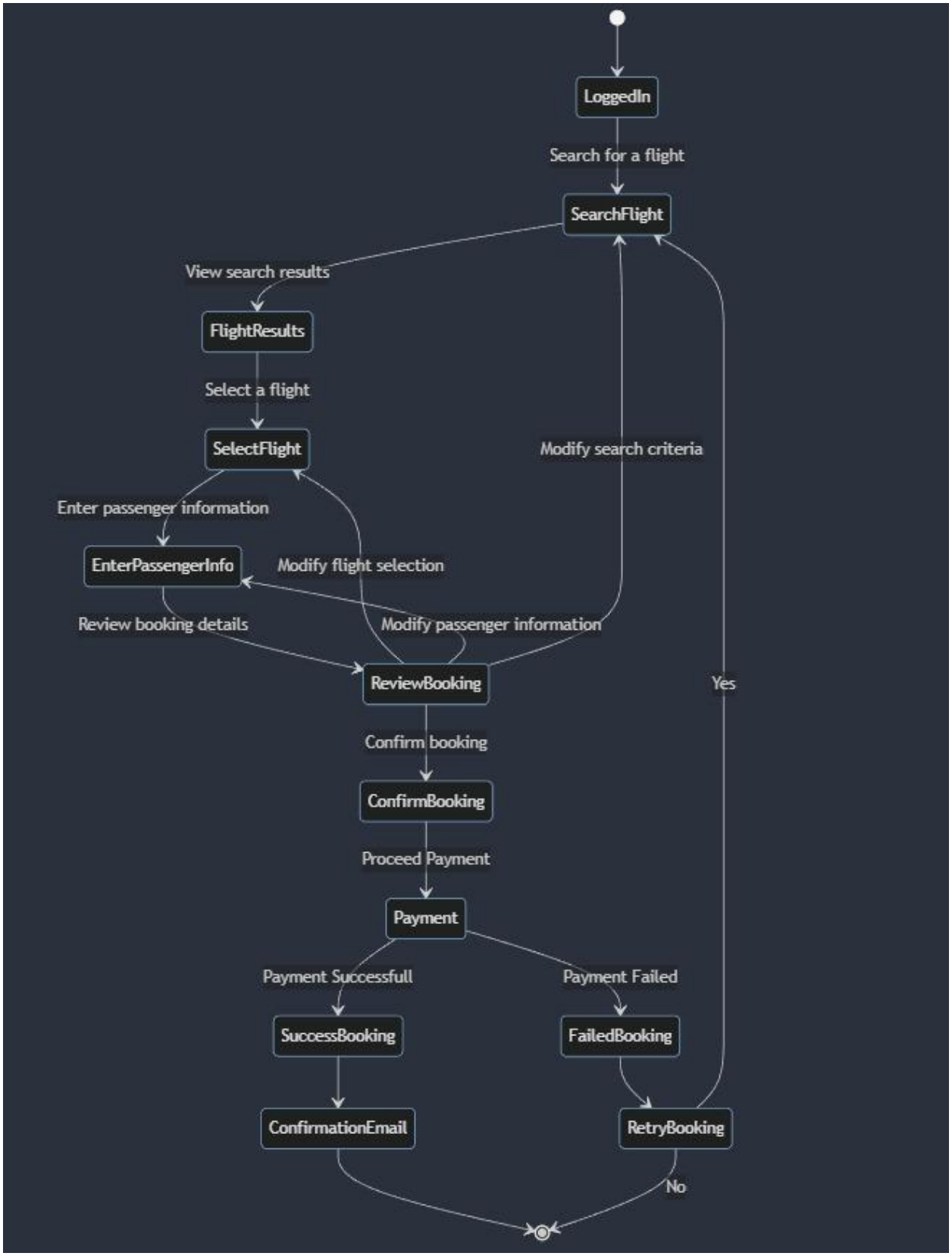


9.1.3 State Diagrams

9.1.3.a State Diagram 1

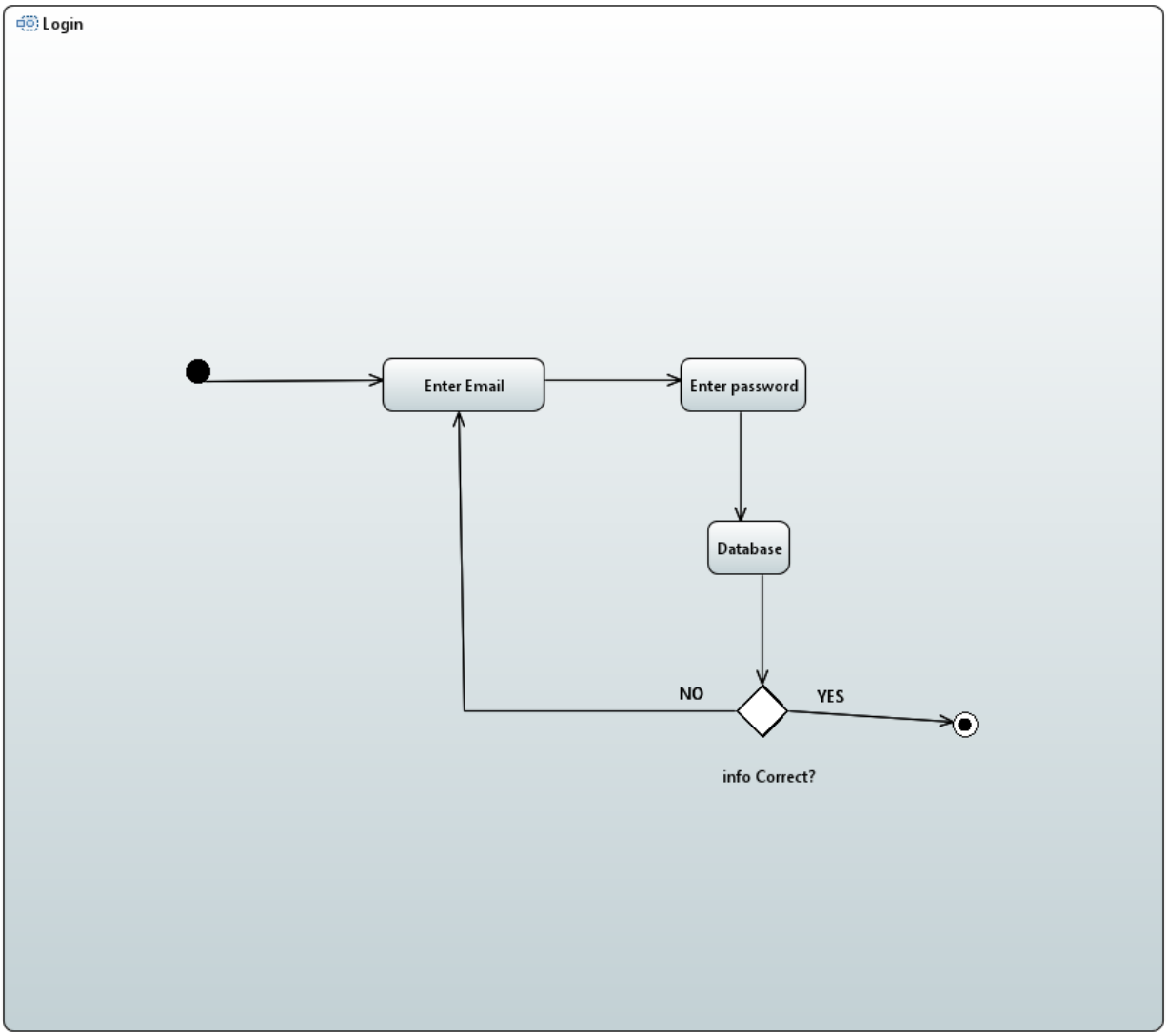


9.1.3.b State Diagram 2

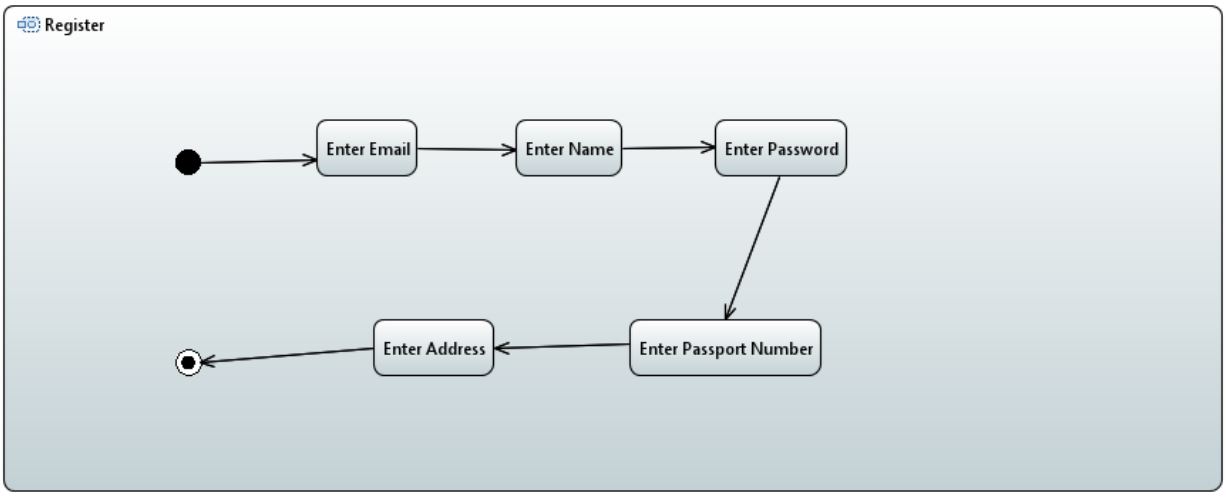


9.1.4 Activity Diagram

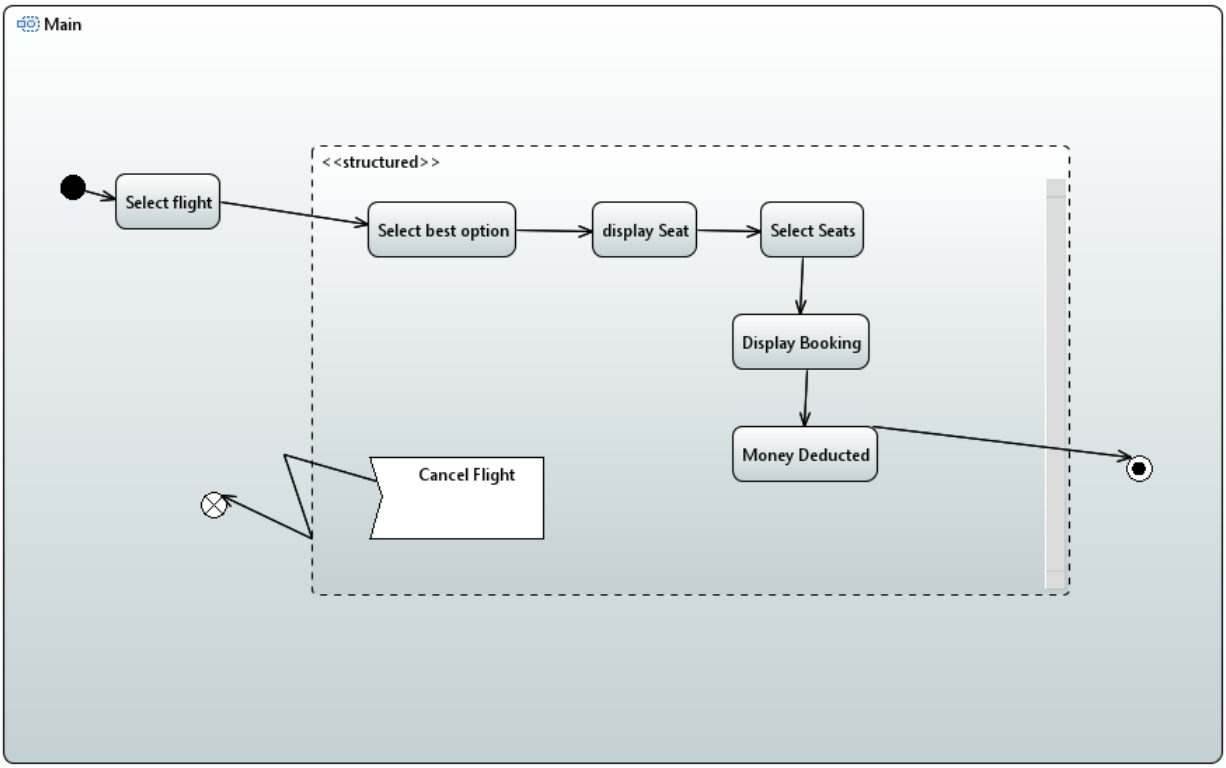
9.1.4.a Activity Diagram 1



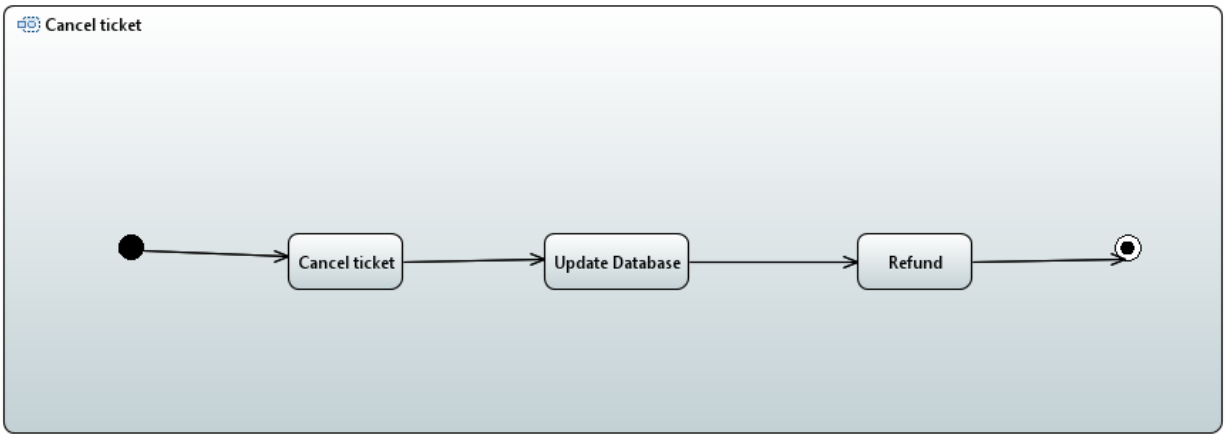
9.1.4.b Activity Diagram 2



9.1.4.c Activity Diagram 3



9.1.4.d Activity Diagram 4



11. Appendices

1. Introduction
2. Overall System Description
3. External Interface Requirement
4. Functional Requirements
5. Non-functional Requirements
6. System Architecture
7. Design Strategies
8. Detailed System Design
9. Application Design