Software Design Document (SDD) Template

Software design is a process by which the software requirements are translated into a representation of software components, interfaces, and data necessary for the implementation phase. The SDD shows how the software system will be structured to satisfy the requirements. It is the primary reference for code development and, therefore, it must contain all the information required by a programmer to write code. The SDD is performed in two stages. The first is a preliminary design in which the overall system architecture and data architecture is defined. In the second stage, i.e. the detailed design stage, more detailed data structures are defined and algorithms are developed for the defined architecture.

This template is an annotated outline for a software design document adapted from the IEEE Recommended Practice for Software Design Descriptions. The IEEE Recommended Practice for Software Design Descriptions have been reduced in order to simplify this assignment while still retaining the main components and providing a general idea of a project definition report. For your own information, please refer to IEEE Std 10161998[1] for the full IEEE Recommended Practice for Software Design Descriptions.

[1] http://www.cs.concordia.ca/~ormandj/comp354/2003/Project/ieeeSDD.pdf

Team 4

Flight company

Software Design Document

Name (s):
Lab Section:
Workstation:

Date: (mm/dd/yyyy)

TABLE OF CONTENTS

1. INTRODUCTION

1.1 Purpose

This software design document describes the architecture and system design of the application for a flight company. Here we will describe all the technical parts that are required for our project to be used.

1.2 Scope

The purpose of the web application for managing a company's flights is to facilitate the booking for the flights of our company for people that want to travel from one point to another with the company planes and all the complementary requirements for booking such as: cancel the booking, details about the date and time for the flight. The administrators have the right to edit the time for the flights or to cancel a flight due to explainable reasons.
Above all, we hope to provide a comfortable user experience along with the best features available.

1.3 Overview

This design requirements document is created to develop a web application which helps its users book flights for a flight company.

- In the first chapter we discussed the overview of the project itself.
- In the second chapter we develop an overview of the system, what is the most important behavior that our site should have, this being how to book a flight.
- In the third chapter we describe the system architecture of the web app. We have chosen the ASP.NET MVC framework to create our website.
- In the fourth chapter we have explained how the elements will be stored in the database
- In the fifth chapter we develop some diagrams about our system functionalities.
- In the sixth chapter we presented how our app should look like and the functionality for each and every page.

1.4 Reference Material

List any documents, if any, which were used as sources of information for the test plan.

Here we will display a list with all the materials that we used in order to develop a plan for testing:

https://www.guru99.com/what-everybody-ought-to-know-about-test-planing.html

https://www.ibm.com/docs/en/elm/7.0.2?topic=testing-developing-test-plans

1.5 Definitions and Acronyms

In this part of the document, we will create a table with the following abbreviations, acronyms and definitions for terms that are used in the creation of the report.

Abbreviations:

Term	Abbreviations
Administrator	Admin
Application	App
Information technology	IT
Information	Info
Departure	Dept
Arrival	Arr
N	Not Added/Not Applicable
IDE	Integrated Development Environment
HTML	Hypertext Markup Language
CSS	Cascading Style Sheets
SQL	Structured Query Language
DB	Database

Definitions:

Term	Definition
User	Someone who uses the application
Administrator	A person that has the role to manage and control the system of the application
Passenger	A person that books a flight in a company plane at a specific moment
Flight	The movement of a plane from a departure point to a destination point

2. SYSTEM OVERVIEW

Our project aims to be a good intermediate environment between a flight company and a client that wants to use our services. So the main page of our application should be displaying a table for choosing the date and cities from where you want to leave, respectively, where you want to arrive. If you are not logged on, the project should send you to the login page. After the user login/register he/she should be able to see a list with all the flights available at that date. After a flight is chosen, the user would be redirected to the price list for the reservations. (Here it should be 3). Once a ticket packet has been chosen the page will redirect the user to the seats selection. If the passenger has chosen the first or second packet it should be able to select he/hers seat without extra charge, if the third option was chosen the seats will be automatically selected by the application. This was the most common scenario a user can use our application.
As another feature the user could select the profile page, where can see his/hers credentials and also could change password or he/she can select the page where all the user bookings should be displayed. Here will be 2 lists, one with already booked one and next flights. For every "already booked" item the user could leave a review.

We have chosen the ASP.NET MVC framework because it allows us to rapidly build robust and scalable enterprise-class web applications, dynamic websites, interactive pages and services. The framework supports HTML/JS, WML and XML, VB.NET, C#, J#, JScript.NET. ASP.NET MVC inherited the strongest sides from ASP.NET — a powerful server-side technology built on the basis of Common Language Runtime (CLR). It is easier to develop, dubbed and deploy and all these actions can be performed within the integrated development called Visual Studio .NET.

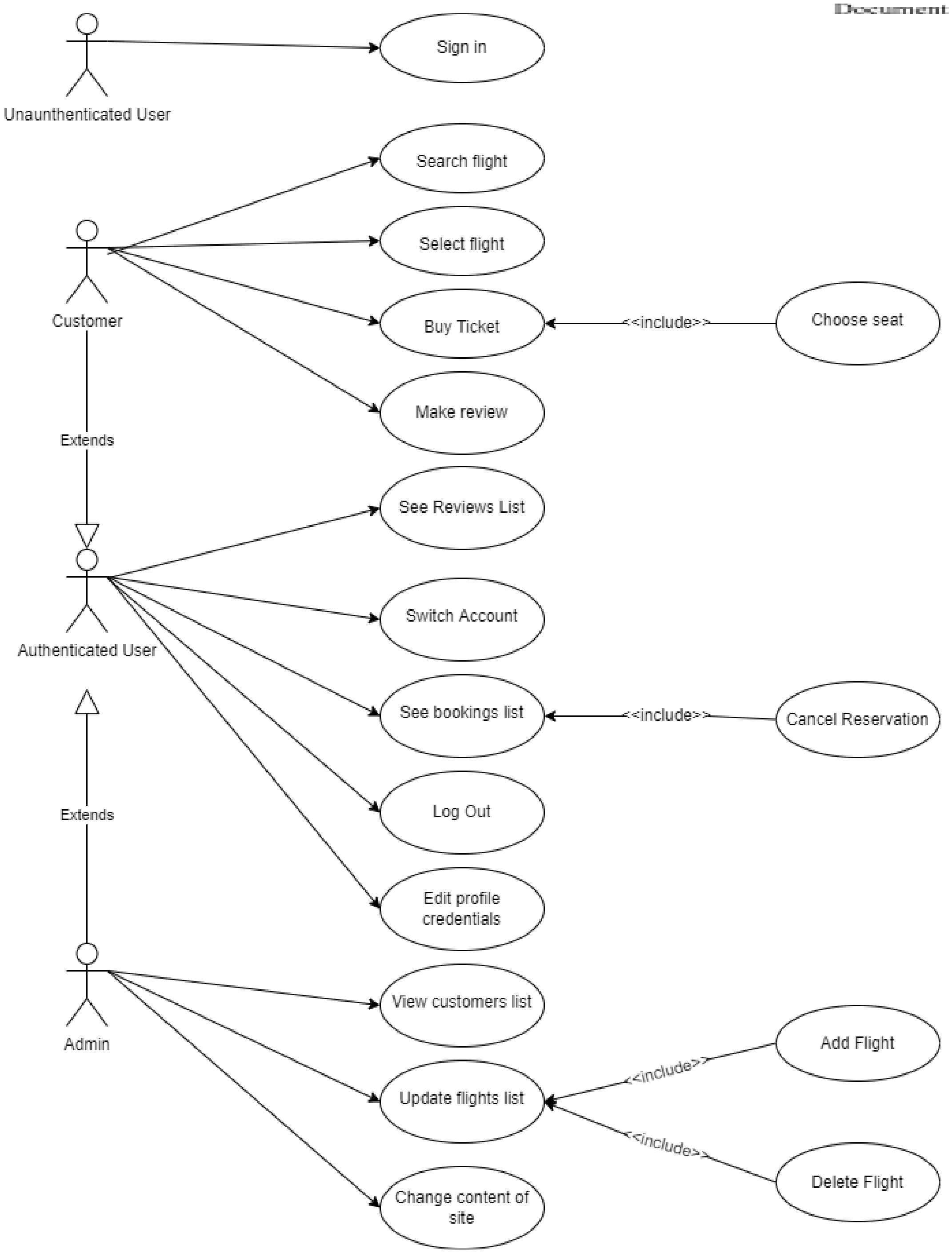
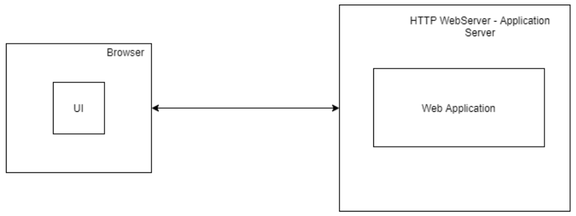


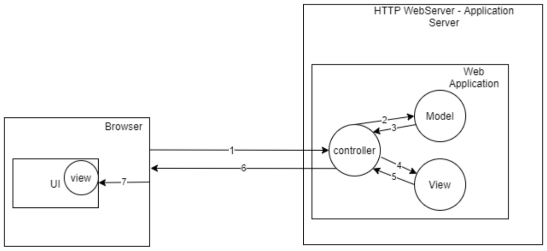
Fig. Use-Case Diagram

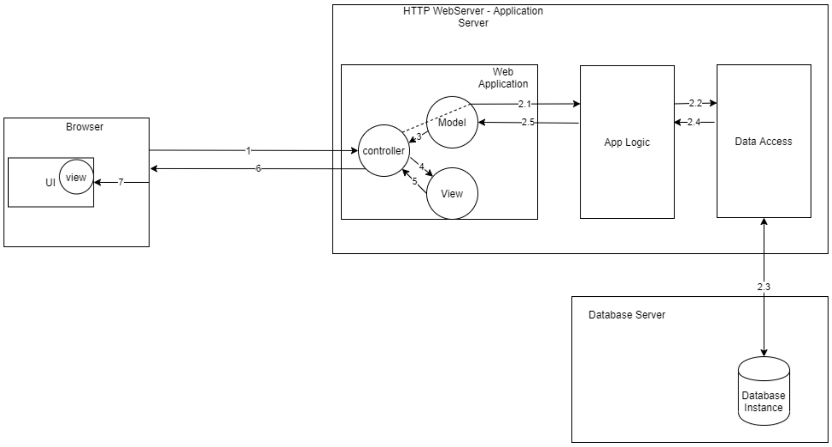
In this **Use-Case diagram** we show all of the user's possible interactions with our system. Our use case diagram shows various use cases and the different types of users that our system has. The use cases are represented by either circles or ellipses.

3. SYSTEM ARCHITECTURE

3.1 Architectural Design







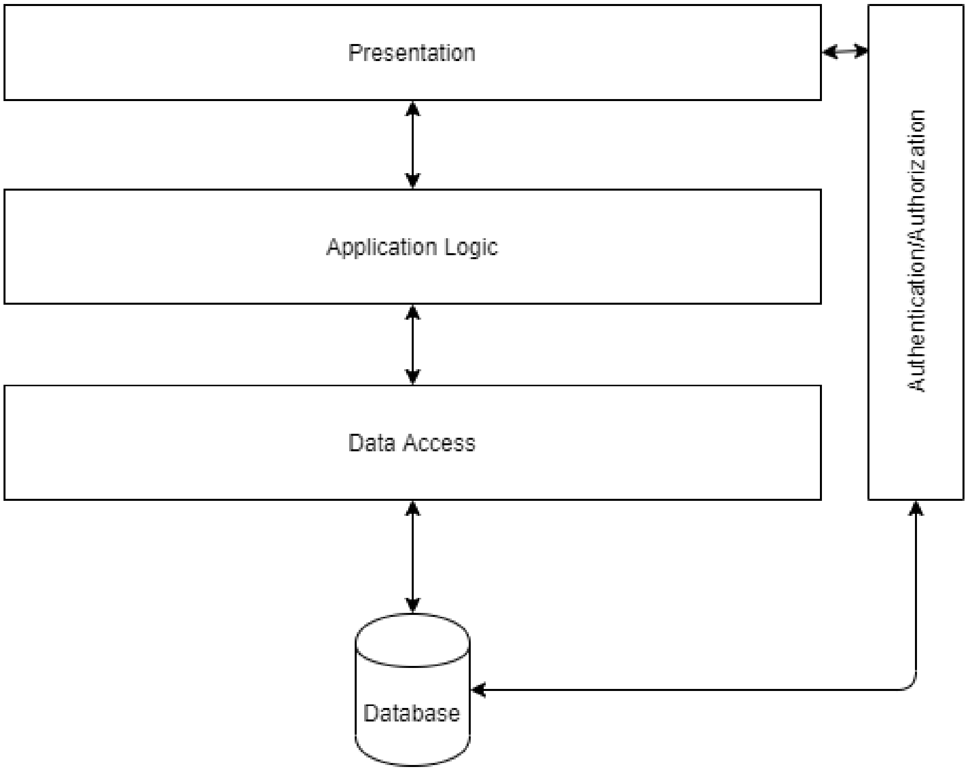
Model, view and controller (MVC) is a well-known three-layer development architecture used for web applications developments. Its first layer is related to the user input logic, second layer is related to the business logic and third layer is used to implement user interface logic. MVC provides very loose coupling among these three layers. MVC patterns are used to specify the location of each logic in application. MVC patterns provide the facility of parallel development.

Many developers use MVC as a standard design pattern. It is a complete framework. MVC provide three types of classes:

Model- Model classes are used to implement the logic of data domains. These classes are used to retrieve, insert or update the data into the database associated with our application.

B. View- Views are used to prepare the interface of our application. By using that interface users interact with our application.

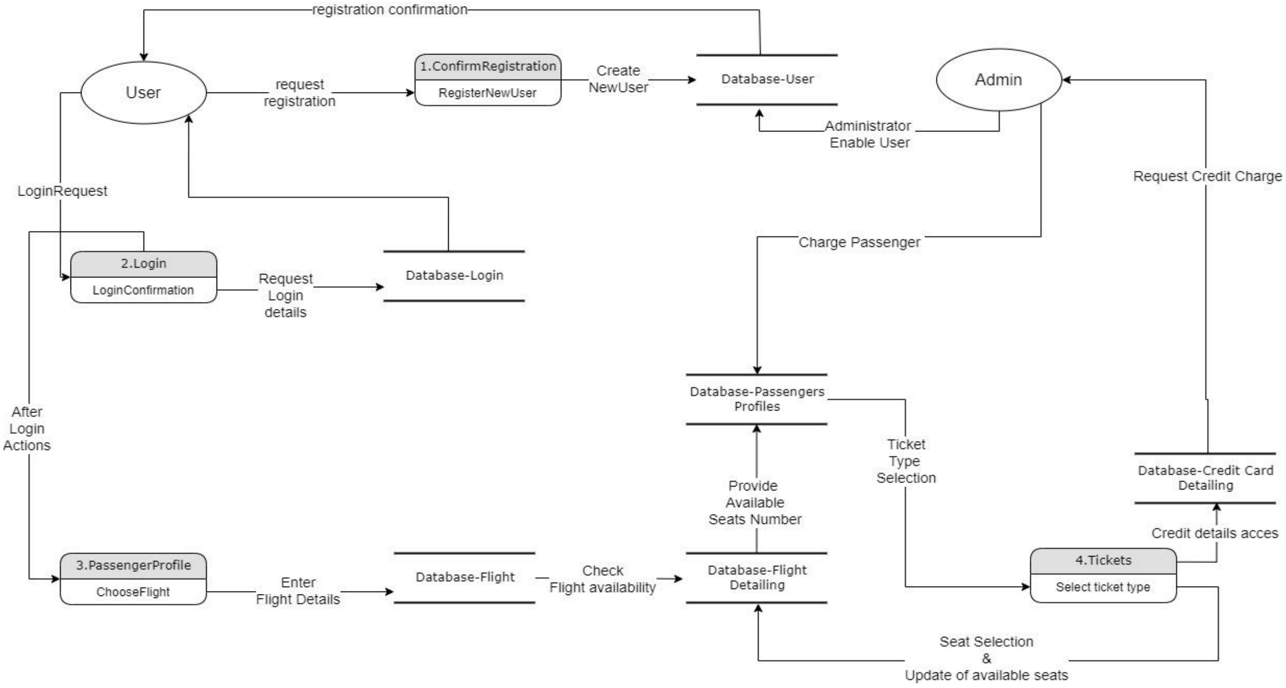
C. Controller- Controller classes are used to respond to the user’s requests. Controller classes perform the user's requested actions. These classes work with model classes and select the appropriate view that should be displayed to the user according to user requests.



As mentioned above, we are developing an MVC application with a layered architecture. In the diagram above we present all these separate layers which communicate with each other. Across all these layers, we have the authentication layer, since a user must first have a certain type of authorization to have access to his/her respective activities, be he/she an admin or a customer of the site.

3.2 Decomposition Description

DataFlow Diagram:



3.3 Design Rationale

As we said before, as our architecture ground, we selected ASP.NET MVC. In the next list of bullet points I will explain why we have chosen this framework:

- **Platform Independence**

  ASP.NET MVC **provides a patterns-based way to build dynamic websites that enables a clean separation of layers**. It gives you full control over markup, supports test driven-friendly development and uses the latest web standards.

- **Language support and development**

  NET supports around 44 languages, and you get the absolute freedom to choose the language you want to employ for the application. One benefit that you receive is that .NET development is a fast process and has library classes giving resources for app development.

- **Cost effective - Better performance**

  .NET is known for its reliability and speed. It has a just-in-time compilation process with better performance and speed. The best part here is that it forms responsive results. You would be with .NET's dynamic experience with the tremendous flexible process.

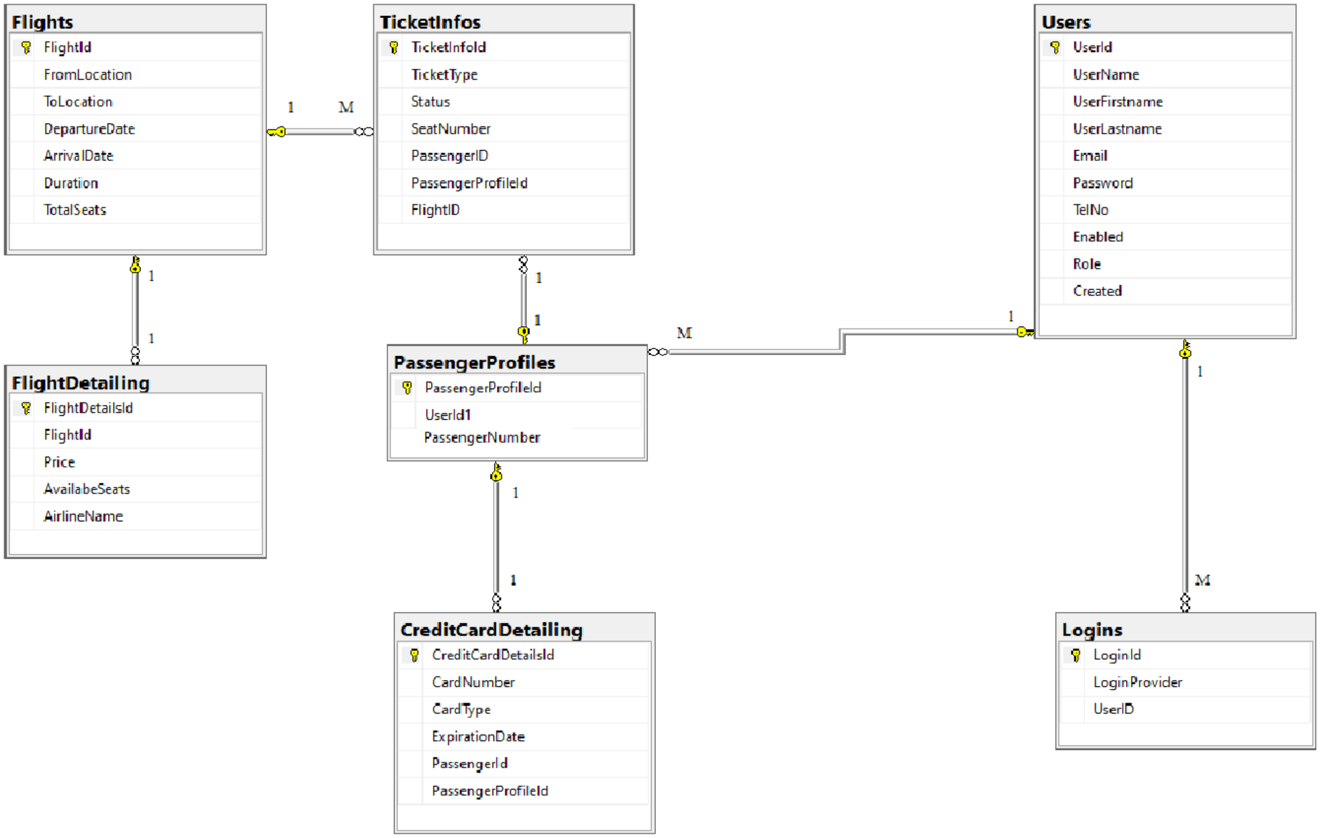
- **Easy access to information**

  Several developers have worked on .NET and have shared their experiences and their challenges between them. They have given their best in answering all the users' queries who get stuck with the entire development process. The community has several developers globally who are there to give you any sort of solution.

4. Data Design

4.1 Data Description

In this section, the relational model is represented using a diagram.
Each table represents the details of the models which will be used in order to shape the database.

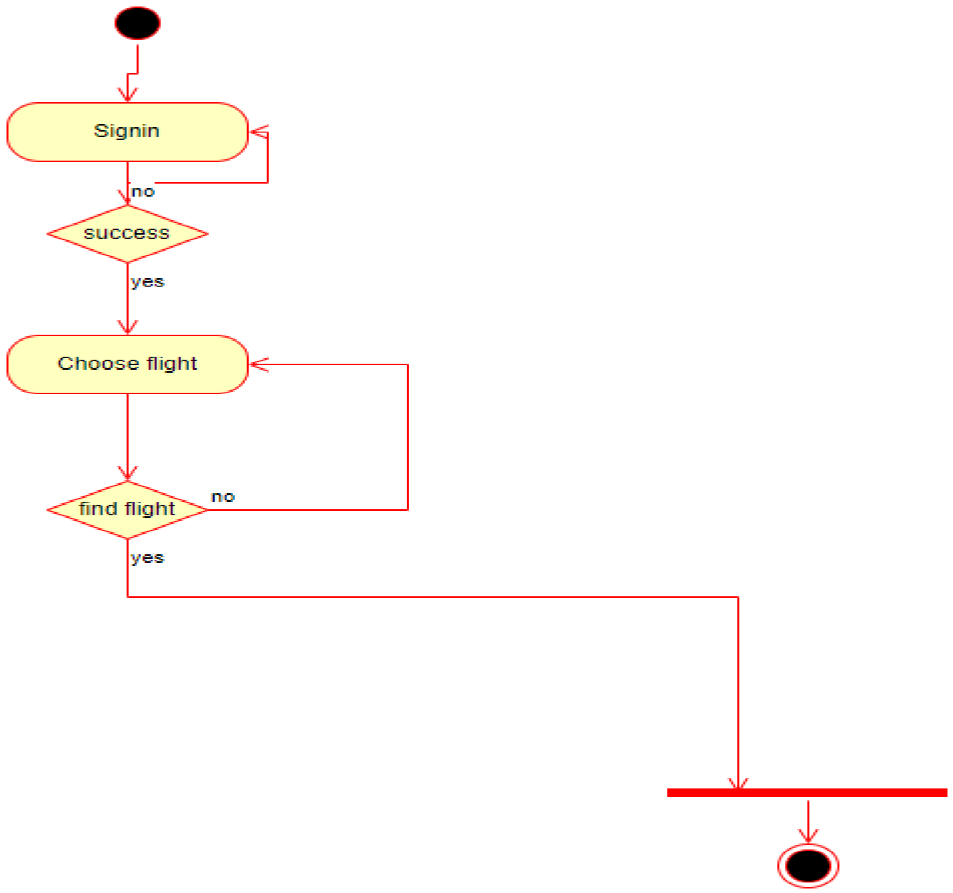


4.2 Data Dictionary

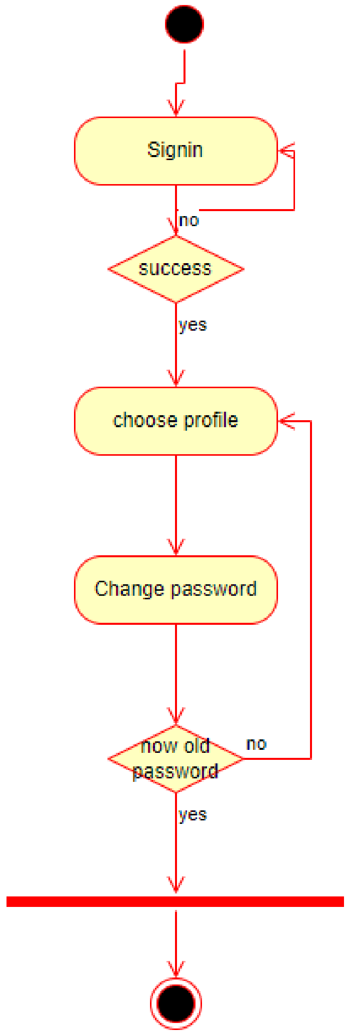
Alphabetically list the system entities or major data along with their types and descriptions. If you provided a functional description in Section 3.2, list all the functions and function parameters. If you provided an OO description, list the objects and its attributes, methods and method parameters.

5. Component Design

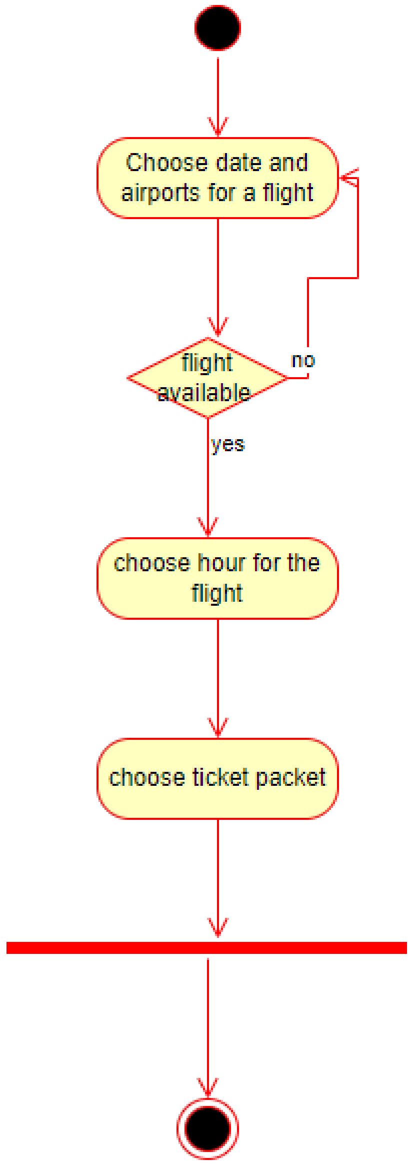
1. choose a flight after you sign in:



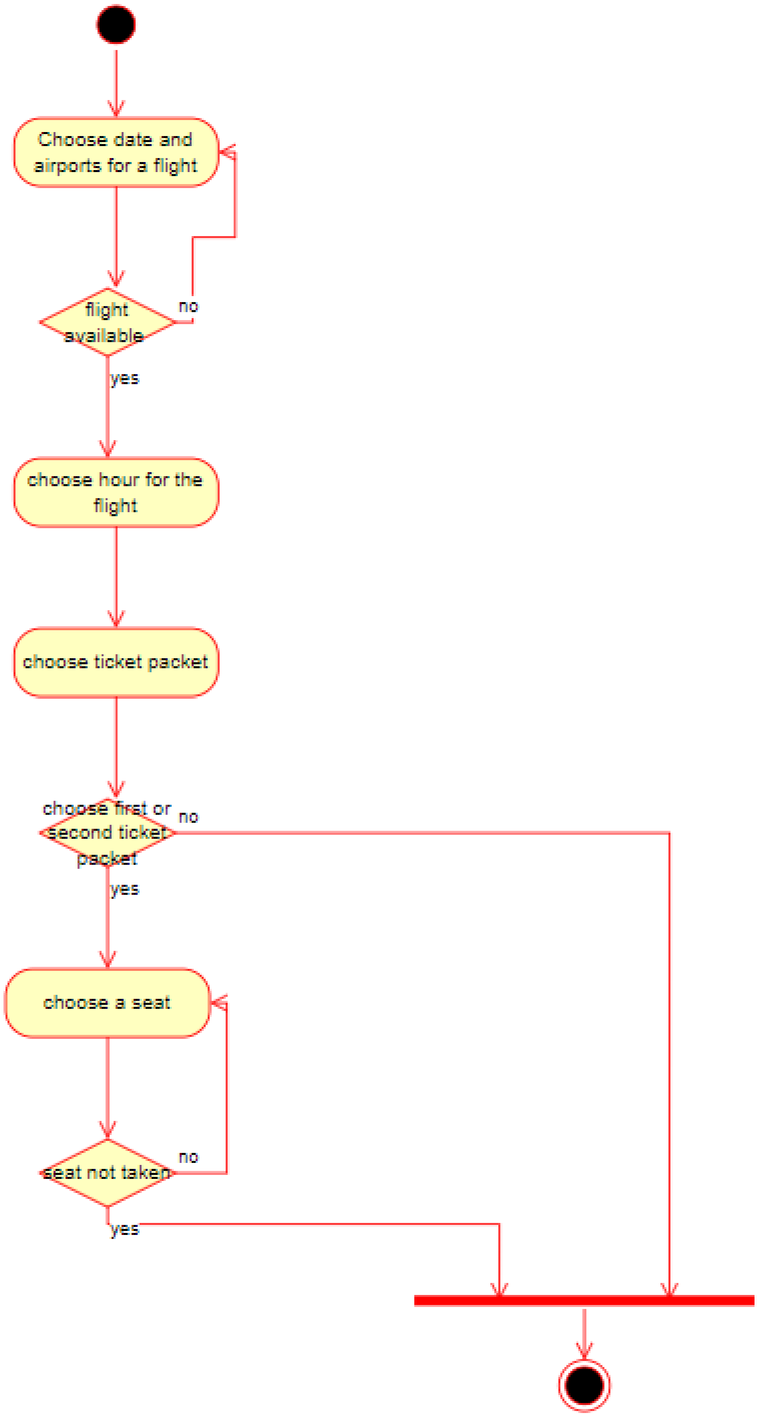
2. change password of an user from profile page



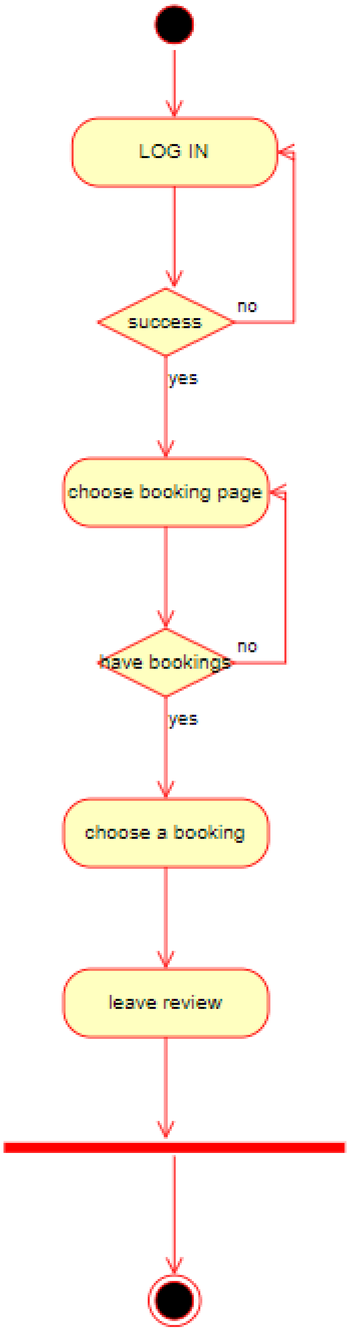
3. choose a ticket for a flight


Choose date and airports for a flight
flight available
no
yes
choose hour for the flight
choose ticket packet

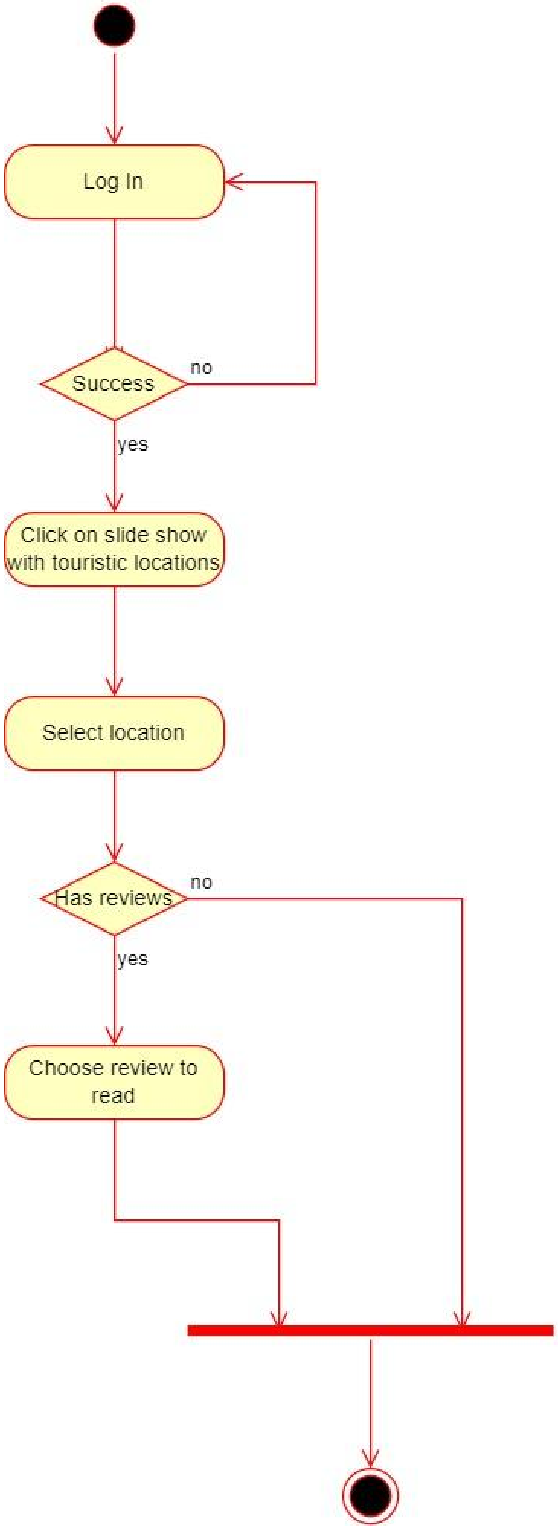
4. choose a seat in a flight


Choose date and airports for a flight
flight available
no
yes
choose hour for the flight
choose ticket packet
choose first or second ticket packet
no
yes
choose a seat
seat not taken
no
yes

5. leave a review



6. see reviews



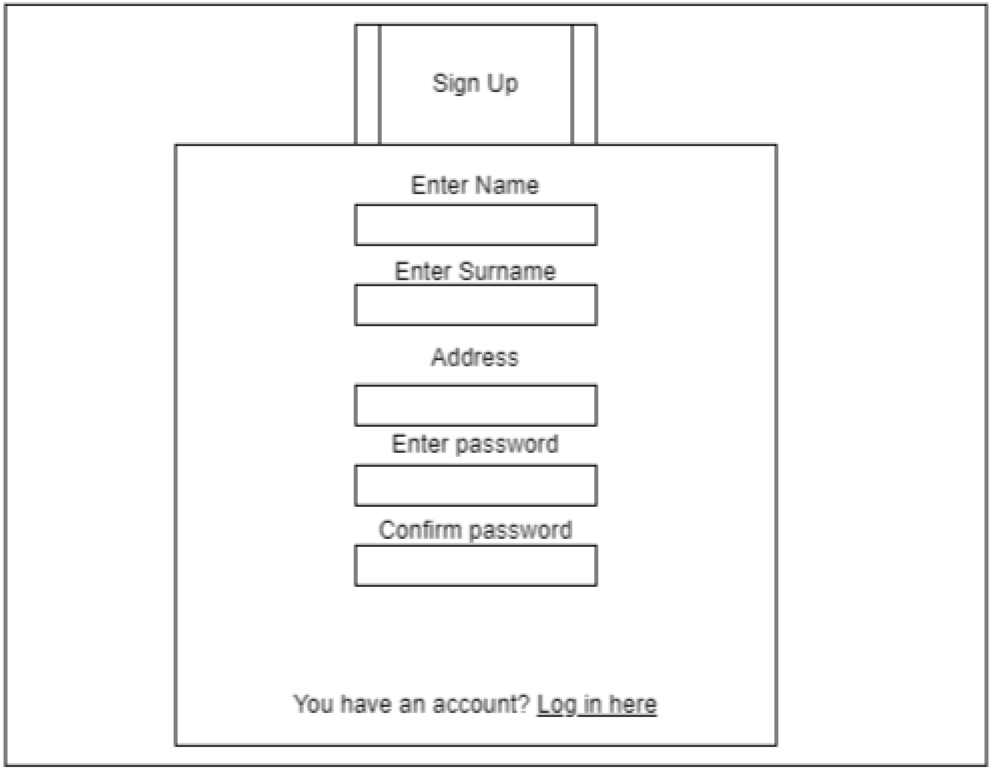
6. HUMAN INTERFACE DESIGN

6.1 Overview of User Interface

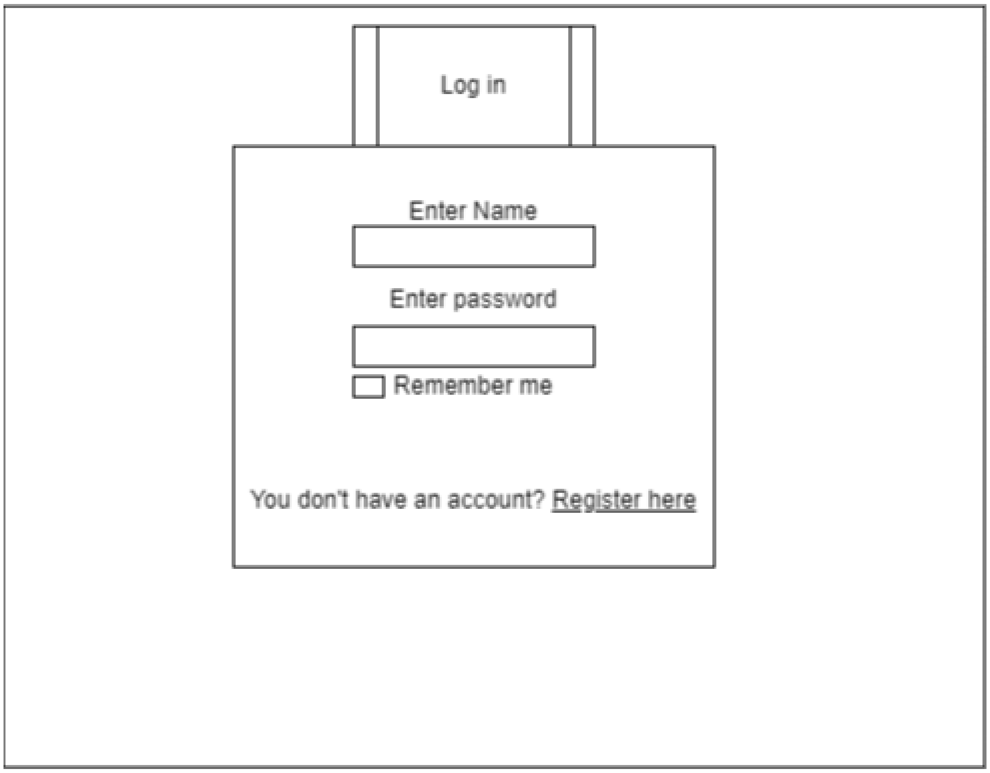
So the main page of our application should be displaying a table for choosing the date and cities from where you want to leave, respectively, where you want to arrive. If you are not logged on, the project should send you to the login page. After the user login/register he/she should be able to see a list with all the flights available at that date. After a flight is chosen, the user would be redirected to the price list for the reservations. (Here there should be 3 ticket options). Once a ticket packet has been chosen the page will redirect the user to the seats selection. If the passenger has chosen the first or second packet it should be able to select he/hers seat without extra charge, if the third option was chosen the seats will be automatically selected by the application. This scenario is when the passenger wants to book a flight.
As another feature the user could select the profile page, where he/she can see his/her credentials and could also change password, or he/she can select the page where all the user bookings should be displayed. Here there will be 2 lists: one with already booked, one with next flights. For every "already booked" item the user could leave a review.

6.2 Screen Images

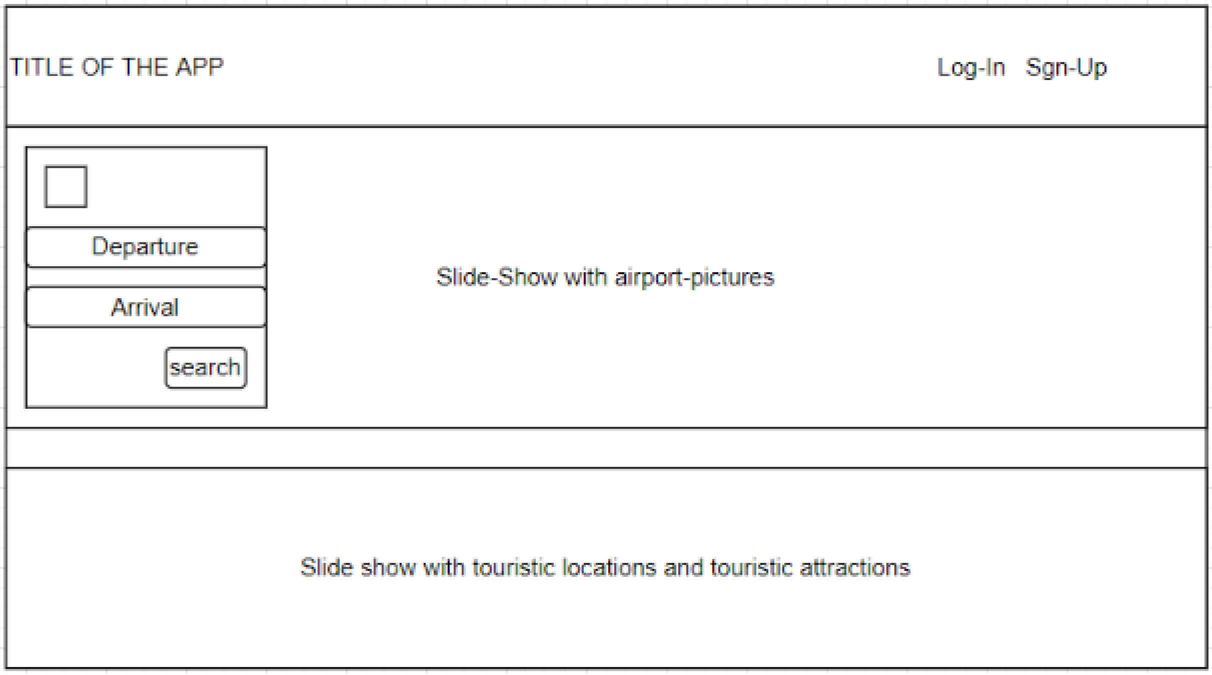
Register Image:



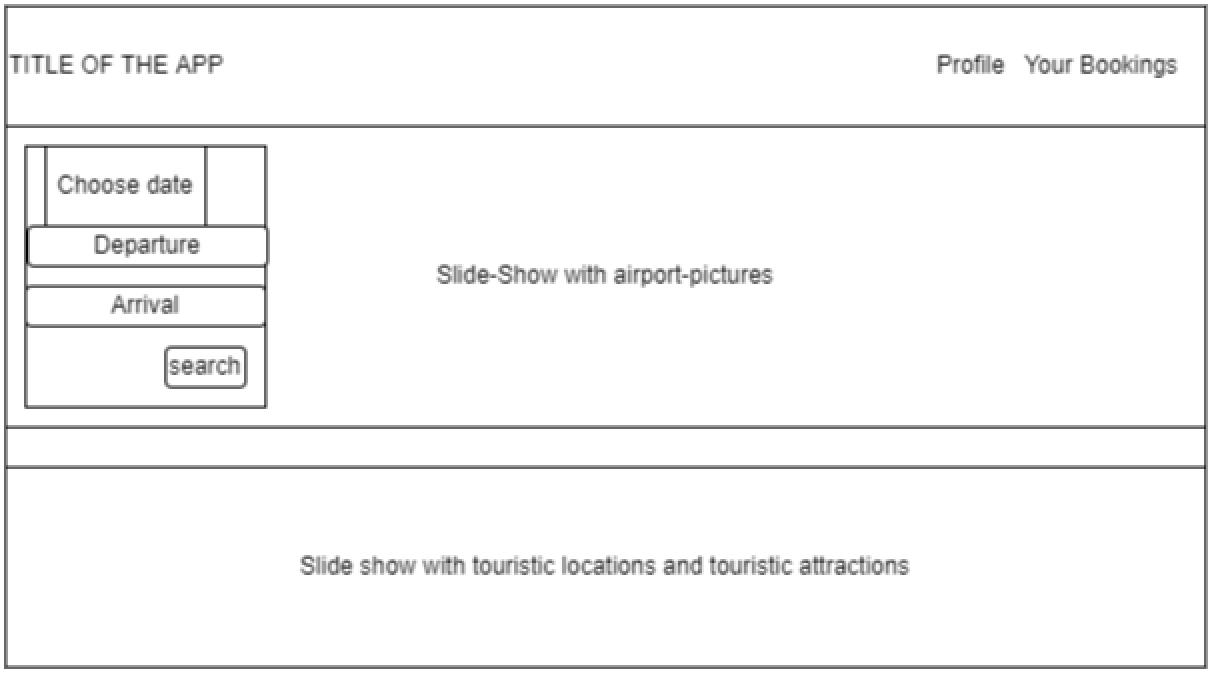
Login page:



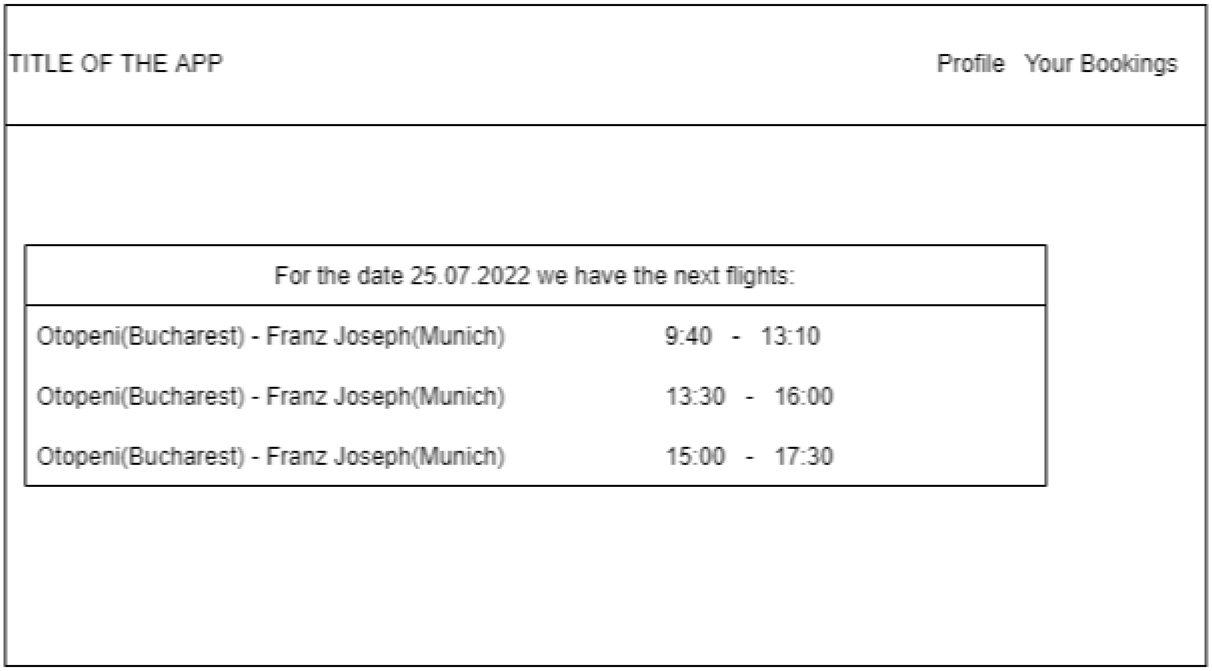
Main page not registered:


TITLE OF THE APP
Log-In
Sgn-Up
Departure
Arrival
search
Slide-Show with airport-pictures
Slide show with touristic locations and touristic attractions

Main page for the user:


TITLE OF THE APP
Profile
Your Bookings
Choose date
Departure
Arrival
search
Slide-Show with airport-pictures
Slide show with touristic locations and touristic attractions

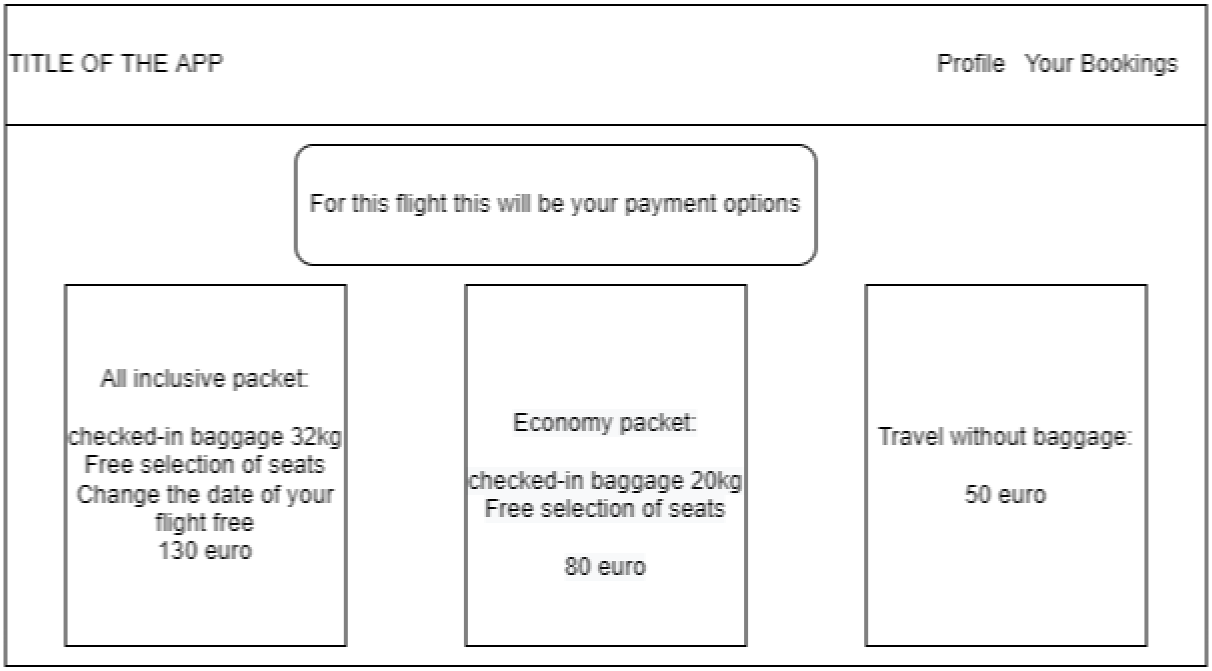
Flight list(appears after the user chooses the date and the airports:



For the date 25.07.2022 we have the next flights:

Otopeni(Bucharest) - Franz Joseph(Munich)	9:40 - 13:10
Otopeni(Bucharest) - Franz Joseph(Munich)	13:30 - 16:00
Otopeni(Bucharest) - Franz Joseph(Munich)	15:00 - 17:30

Price list (appears after the passenger selects a flight):



TITLE OF THE APP

Profile Your Bookings

For this flight this will be your payment options

All inclusive packet:

checked-in baggage 32kg
Free selection of seats
Change the date of your flight free
130 euro

Economy packet:

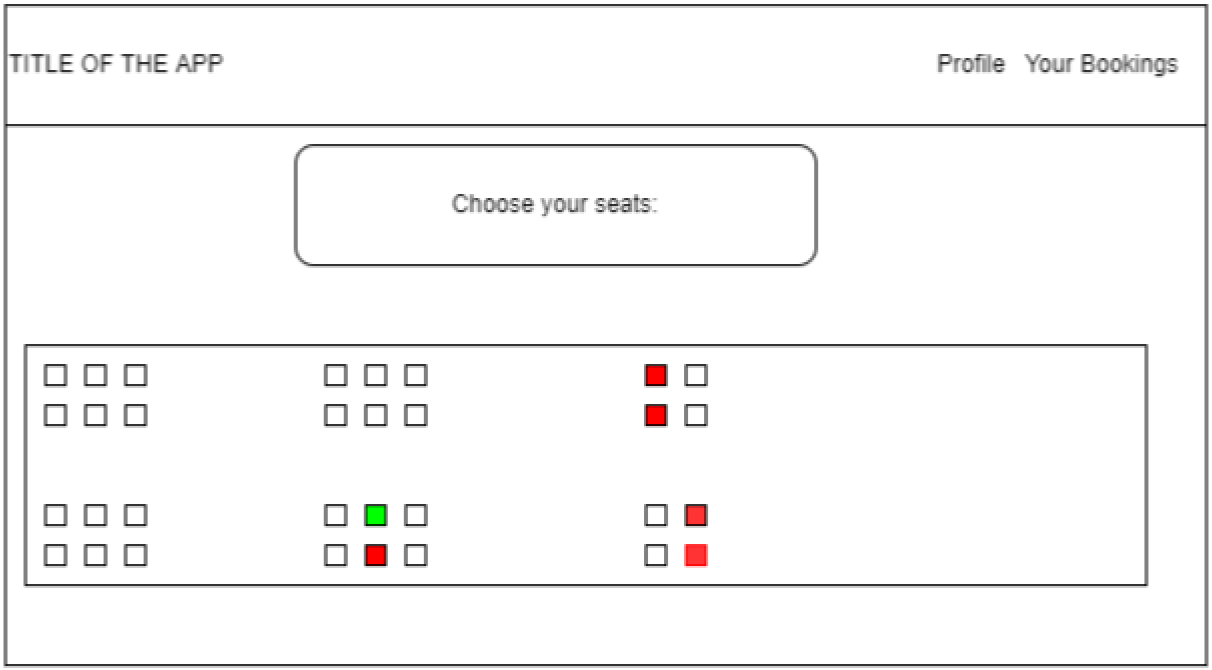
checked-in baggage 20kg
Free selection of seats

80 euro

Travel without baggage:

50 euro

After the ticket packet is selected the selection seats page appears:

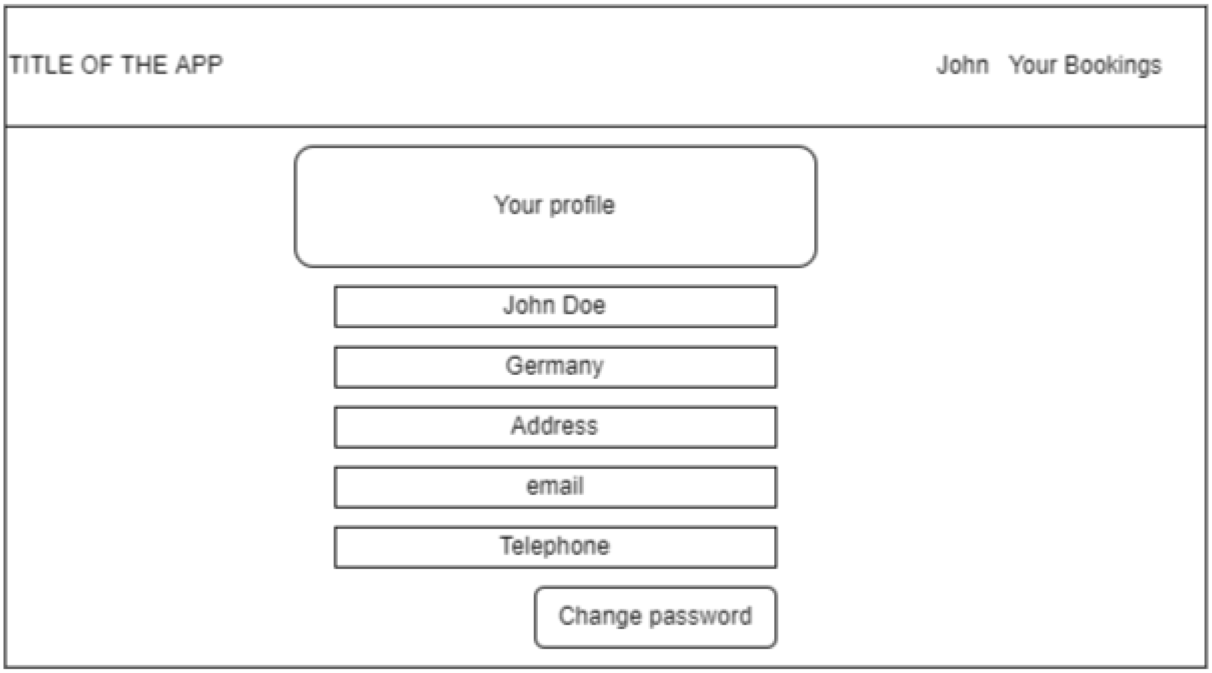


TITLE OF THE APP

Profile Your Bookings

Choose your seats:

Profile page:



John Doe

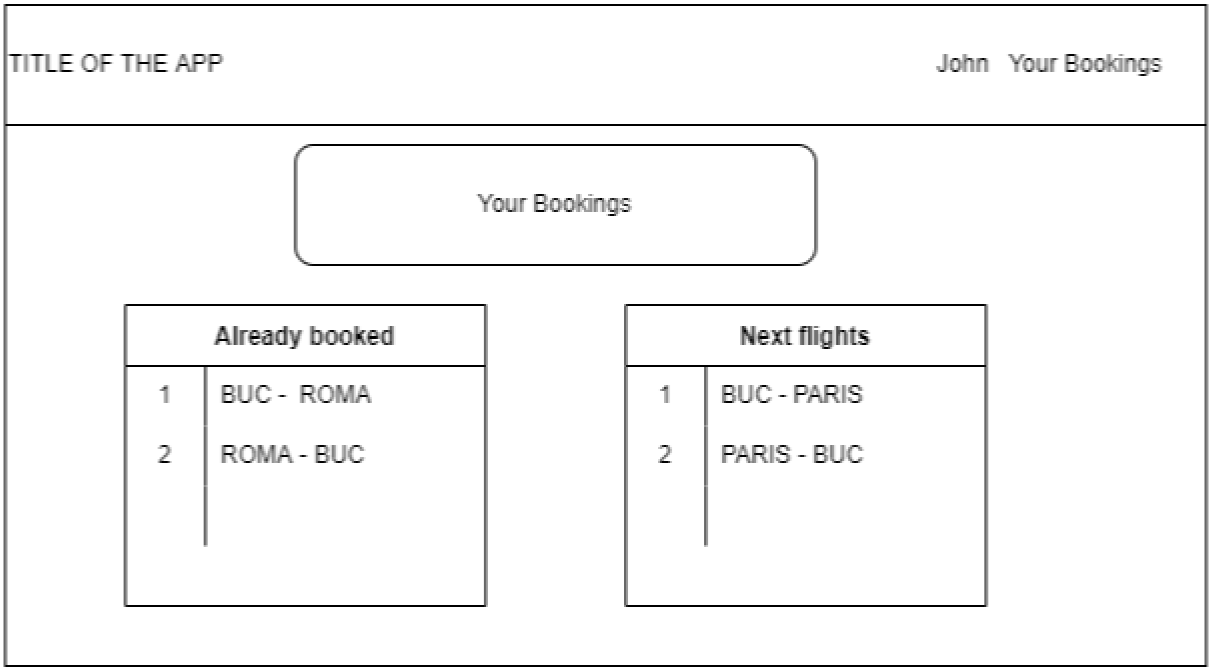
Germany

Address

email

Telephone

Booking page:



Already booked	
1	BUC - ROMA
2	ROMA - BUC

Next flights	
1	BUC - PARIS
2	PARIS - BUC

6.3 Screen Objects and Actions

- **Register page**
  In this page an user can create an account to use our app. For this he/she needs to fill some text boxes in order to add their credentials for a flight such as: name, surname, address and password. After the user manages to fill all the textboxes that he/she can hit the register button that will save all his credentials in the DB .If the user already has an account he/she could switch to the login page using the link in the bottom of the form.
- **Login page**
  In this page the user can log into the application. For this he/she needs to fill the email and the password of the account. After that the login button is pressed. If it is a user who already has an account and the identification elements are written correctly the main page should appear. If the user does not have an account he/she could switch to the register page using the link in the bottom of the form.
- **Main page for users**
  In this page the user can see the navbar of the application. This is formed with the title of the app,and to other buttons, *profile* where the user can change the password and see the credentials and *your booking* where the user can see the booking that they already had and leave a review. The main page consists in a table where the user can choose the date of the flight and the airports. The design of the page is maintained by 2 slide shows. One with airport pictures and another with touristic locations
- **Flight list**
  After the user selects the flight date and airports a flight list should be displayed. The page also has the navbar of the main file. The flight list consists of the names of the airports from the locations and also the departure and arrival times. Once the user decides which flight is convenient in the form of time it selects the respective flight.
- **Tickets price list**
  When a flight is chosen, the price list will be displayed. This consists of 3 packets: All inclusive, economy packet and travel without baggage. These 3 options come with their description displayed in the form of a cardboard. The all inclusive one has checked in baggage that consists of 32kg, free selection of seats which will be discussed in the next topic, and also the feature that the date of the flight could be changed for free. The economy packet has checked in baggage of 20 kg and also the selection of seats. And the last one has no extra benefits. The page also has a navbar with the title profile and booking buttons.
- **Seat selection**
  In the seat selection page a user that purchases an all inclusive packet or economy packet could select their seat in the plane. All the white seats are available for

booking, the red ones are the seats that have already been booked. When a user selects a white seat, it becomes green and the coordinates of the seats are displayed on the page. When at least one seat has been selected an *finalize booking* appears.One is pressed the booking process has been finished and the booking added in the DB.

- **Profile page**
  In the profile page the user could see his/hers credentials. It will be displayed as a form with all the elements added in the sign up process. A feature in this page is that the user could change the password. by clicking on the change password button. Here he/she will be redirected to the page where they should add the last password and add the new one. When all the security conventions are checked the page will give a validation message that the password has been changed. After this process he/she will be redirected to the profile page. the navbar changes when this page is accessed. the profile button becomes the name of the user.
- **The booking page**
  The booking page can be accessed from the navbar using the button *your bookings.* This page consists of 2 lists: one with all the booking that has already been booked and one with all the flights that are in progress. When a user selects a booking, they can write a review about the flight and the flight attendants. On the other list the user could change the date of the flight if the passenger has an exclusive packet.

7. Requirements Matrix

-

8. Appendices

-