Name:	Edwin Rodriguez (ED04)
ID:	Mustafa Abbasi (D07)
Project:	Mini-Project 1
Class:	CMPUT 291
Due Date:	Oct 27, 2015

No Collaboration was done for this project.

C291G40’s Airline Booking program:

The Airline Booking program is an extensive online tool that allows users and airline agents access to their airline booking account. The program has two main functions. First it allows users with the ability to search for direct or round-trip flights. Secondly it provides users with the ability to book flights, view a previous bookings and cancel bookings. In addition to this functionality, airline agents have the ability to update the actual arrival and departure times of a flight which might occur due unforeseen circumstances.

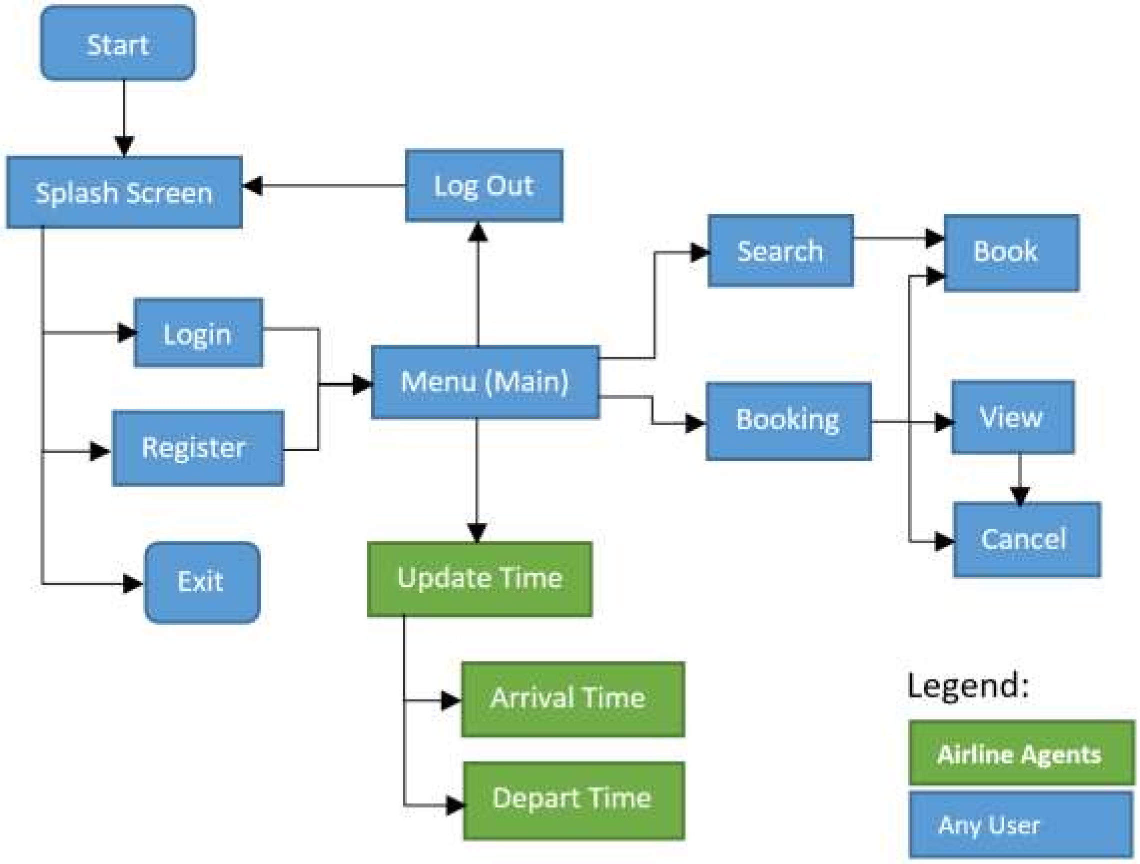

Figure 1: Structural Schematic of the Airline Booking program.

Program Walkthrough:

The program comes with an easy to use menu. To select an option, simply enter the number next to an option.

1. To launch the program launch the following command:

```
Python3 main.py DatabaseUSerName DatabasePassword
```

2. Once the program has been launched the user has 3 options:
    a. Login: You will be prompted for your email address and password.
    b. Register: You will be asked for an email address (20 characters or less) and a password (4 characters or less. Passwords are **case-sensitive**)

   c. Exit: Exits the program.

3. Once logged in or registered, you will be brought into the main menu. All users will have the following options:
   a. Search for flights: This will ask users for the basic information required to search for a flight. Users have the ability to sort flights by flight and differentiate between direct and non-direct flights). Lastly, users will be prompted with an option to book a flight from the search results.
   b. Book a Flight: This allows users to book a flight if they know the flight number, fare type, and the departure date. It can also be used to book round-trips.
   c. View All Bookings: This lists all the bookings belonging to the user. The user can view more details by selecting "V" and the option number, the user can cancel a booking by selecting "C" and the option number, or press "Q" to return to the main menu.
   d. Cancel A Booking: This will list all the flights belonging to the user and the user will be prompted to cancel a booking based on the results of that list.
   e. Logout: This will logout the user and record the time.

   Once an option a, b, c, or d are completed, the user is taken back to the main menu. If a user chooses to logout, the user is re-directed to the login/register screen.

   Airline Agents will be presented with two additional options:

   f. Record flight departure: This will prompt the agent for flight information and it will update the actual departure time of the flight.
   g. Record flight arrival: This will prompt the agent for flight information and it will update the actual arrival time of the flight.

Detailed Design of Software:

The detailed design of your software should describe the responsibility and interface of each primary class (not secondary utility classes) and the structure and relationships among them. Depending on the programming language being used, you may have methods or functions instead of classes.

The program is composed of the following classes modules:

1. main.py:
   a. Responsibility:
      The responsibility of this class is to create the main menu and create the contents of the menu based on the type of user that logged in.
      It also ensures that once users are logged in to the system, they remain or return to this main menu until they decide to logout.
   b. Interface:
      As this is the main arterial of the program, this interfaces with the other 4 components. This interaction is done within the processMenuSelection method.
   c. Relationship:
      Hub to the entire program.
2. splashScreen.py:
   a. Responsibility:

To direct log in existing users or register new users.
To give the program with sufficient knowledge so that the program may differentiate between normal users and airline agents.

b. Interface:
The start() function starts the splashpage and returns true once a user logged in.
getEmail() returns the email of the user
getPassword() returns the password of the user
isAgent() returns true if the user is an airline agent.

c. Is accessed by Main.py only

3. booking.py:
   a. Responsibility:
   Responsible for making, viewing and canceling all bookings.
   b. Interface:
   makeBooking(email): begins the process of making a booking
   cancelBooking(email): begins the process of canceling a booking
   listExistingBookings(email): begins the process of listing all bookings
   c. Is accessed by Main.py only.
4. search.py:
   a. Responsibility:
   Responsible for searching for all flight information.
   b. Interface:
   Main(email, connectionString): starts the process of searching for flights
   c. Is accessed by Main.py only.
5. updateTime.py:
   a. Responsibility:
   Responsible for updating actual arrival and departure time of scheduled flights.
   b. Interface:
   updateTime(connectionString, TypeOfFlight): Updates the actual departure or arrival time of a scheduled flight. TypeOfFlight is a string containing “Arrival” or “Departure”.
   c. Is accessed by Main.py only.

Testing Strategy:

The program was developed under control database conditions where the database had a small data sample. This allowed us to manually verify the output of each of the queries required by the program. Manual verification of the contents of the database was also done to verify updates or deletions from the database that occurred in the program.

Basic input and output from the program was also tested to insure that the program behaved as expected. This led to the development of error checking methods for data inputted from the user. This can mostly be seen in the case of the splash menu, the user menu and date entries into the program.

There was also integration testing which occurred to insure that the program worked after combining all the modules/classes together.

The tests had a small data coverage overall and regression testing ensured that the program maintained its working integrity after big changes occurred.