# Assignment 2: Abstract Classes, Event Driven Applications

**Scenario**

A local travel agency, Traveless, has hired your company to implement a new flight reservation management system to improve its productivity and services. After meeting with the agency, you realize that to tackle this complex request, you must break down the solution to three major parts:

- Front-end GUI
- Backend
- Functional front end connected to the backend.

**Equipment and Materials**

For this assignment, you will need:

- Visual Studio IDE
- Supplied project skeleton, project hierarchy, and data files
    - Download from the BrightSpace (D2L)
    - Data files
        - flights.csv
        - airports.csv

**Instructions**

1. Review the scenario, and then carefully read the Traveless program Details and Program Guidelines sections of this document.
2. Working outside of class time, complete the submission sections of this assignment.
3. Review the grading criteria for the group submissions.
4. See the course schedule and/or Brightspace for due dates.

**Submission**

1. Create the code for a program that meets the requirements described bellow.
2. Test your code against the expected output provided.
3. Check your program against the detailed marking criteria at the end of this document.
4. Submit the following to Brightspace as a group (Only one copy is required per group, and any of the group members may submit):
    - Github URL for your program code (invite your instructor to be a member of the project repository) (Optional)
    - Screenshots for different output

**Peer Assessment (5%)**

Each student must also complete a peer assessment of their group members. Your instructor will provide further submission details.

## Traveless Details

Create a functional, event-driven program that manages the travel agency's data (contained in the provided data files) and allows a user to do the following:

### Find Flights

- A travel agent can find a flight by providing the origin airport, the destination airport and the day of the week the flight departs.

### Make a Reservation

- A travel agent can make a flight reservation for a traveller.
- A reservation code is generated and assigned to the traveller's name and citizenship.

### Find Reservations

- A travel agent can find existing flight reservations using the reservation code, and/or the airline and/or the traveller's name.

### Cancel an Existing Reservation

- A travel agent can cancel a flight.

## Detailed Requirements

When the graphical user interface is launched, the user can choose to either search flights and make a reservation, or search for and modify a reservation.

### Find Flights

The *findFlights()* in the *FilightManager* method receives as its input arguments: the **originating** airport, the **destination** airport, and **the day** of week. The method returns a List of any **matching *Flight* objects**. If no matches are found, the list control is empty.

### Make Reservation

When a travel agent selects a flight from the list, the text fields are populated with the selected flight code, airline, day, time and cost. The travel agent enters the traveller's full name and citizenship. **The flight code, airline, day, time and cost cannot be edited**. An **error message is displayed if:**

- **A reservation is to be made but no flight is selected**
- **The name field is empty**
- **The citizenship field is empty**

### Find Reservations

A travel agent can **search for an existing reservation that contains the specified reservation code, and/or airline and/or traveller's full name**. The list is populated with any reservations that are found. Each row in the list displays the code of the corresponding reservation record.

### Cancel Reservation

When a reservation in the list generated by the findReservation method is selected, the corresponding, the user can select any one of the list and cancel it. The status for the flight should change from Active to Cancelled.

### Steps

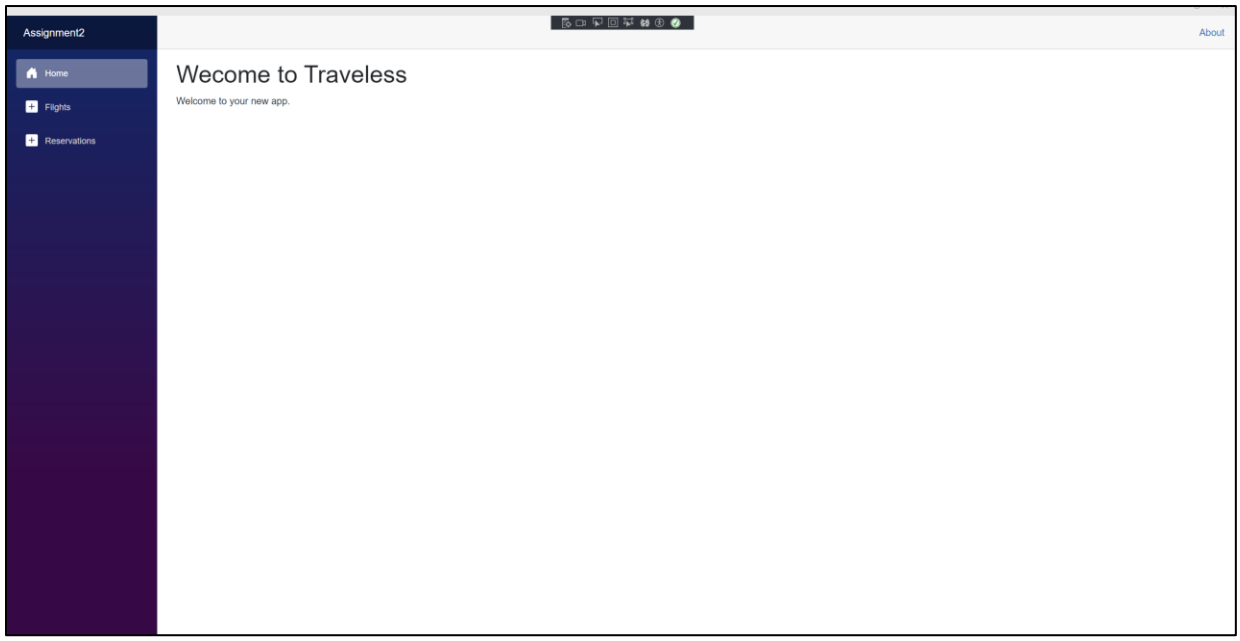
- Update the UI as you need.
- Complete all TODO parts in the code to get the proper functionality from the project.

**Notes**

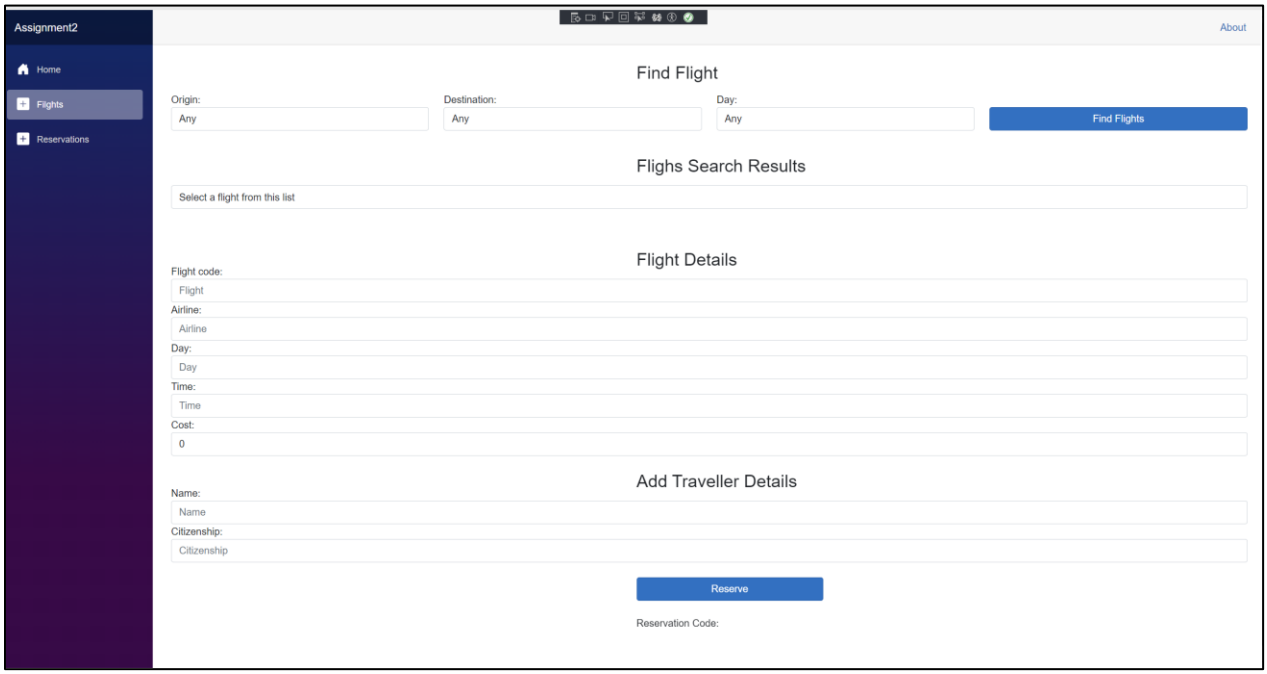
- The *ReservationManager* class generates the reservation code.
- The name and citizenship do not need to follow any specific format; however, they cannot be empty.
- Each problem domain class overrides the *toString()* method.
- Flight codes use the following format: (L meaning letter, D meaning digit)
    - *LL-DDDD* (e.g.: *GA-1234*)
- Reservation codes use the following format: (L meaning Letter, D meaning Digit)
    - LDDDD (e.g., *I1234*)
- Times are in 24-hour format: HH:MM
- A reservation that is set to inactive is persisted and retained when the program opens again.

## Samples of the Run

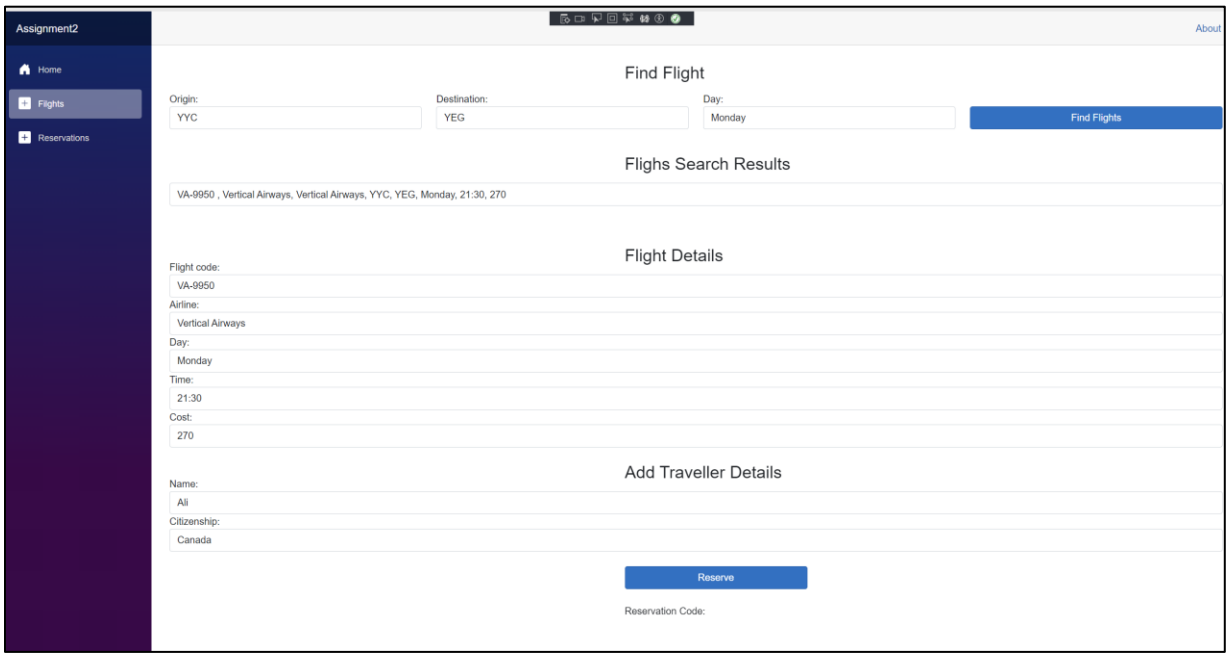
Home Page




Flights Page




Flight Page with data




Reservation Page

## Marking Criteria

### Peer assessment/Contribution list Submission

| | Not submitted (0%) | Submitted (100%) | Marks |
|---|---|---|---|
| **Peer assessment** | • Not submitted | • Completed for the group member | **/5** |

### Group Submission

| | Needs Improvement (0–50%) | Good (51–75%) | Excellent (76–100%) | Marks |
|---|---|---|---|---|
| **Working Code** | • The project doesn't run in all scenarios<br>• Syntax errors | • The project runs in all scenarios | • The project runs in all scenarios | **/10** |
| | • Input requests don't work | • Input requests work but don't match the scenario | • Input requests match the scenario exactly | **/10** |
| | • Output works but doesn't match the scenario | • Output works but doesn't match the scenario in all cases | • Output matches the scenario perfectly | **/10** |
| | • User interface design and functionality requires improvement | • User interface design and functionality works, but there is room for minor improvement | • User interface design and functionality works perfectly and is thoughtfully designed | **/15** |
| | Many of the following items don't work:<br>• Populate airports and flights<br>• Find flights<br>• Generate a reservation code<br>• Make a reservation | Some of the following items don't work:<br>• Populate airports and flights<br>• Find flights<br>• Generate a reservation code<br>• Make a reservation | All of the following items work correctly:<br>• Populate airports and flights<br>• Find flights<br>• Generate a reservation code<br>• Make a reservation | **/20** |

| | Needs Improvement (0–50%) | Good (51–75%) | Excellent (76–100%) | Marks |
|---|---|---|---|---|
| | Many of the following items don't work:<br>• Find reservations<br>• Store reservations in a random-access file<br>• Cancel a reservation<br>• Populate lists and allows selection | Some of the following items don't work:<br>• Find reservations<br>• Store reservations in a random-access file<br>• Cancel a reservation<br>• Populate lists and allows selection | All of the following items work correctly:<br>• Find reservations<br>• Store reservations in a random-access file<br>• Cancel a reservation<br>• Populate lists and allows selection | **/10** |
| **Style** | • Indentation – not consistent<br>• Readability – poor variable names<br>• Documentation<br>○ No comments are included at the top.<br>○ No comments indicating major code sections or what they do | • Indentation – mostly consistent<br>• Readability – some variable names are not ideal<br>• Documentation<br>○ Comments at the top are missing or incomplete.<br>○ Comments indicating major code sections and what they do are incomplete | • Indentation – consistent<br>• Readability – good variable names<br>• Documentation<br>○ Comments at the top are complete and include name, date, program description including details on inputs, processing and outputs (4–5 sentences minimum).<br>○ Comments indicate major code sections and what they do | **/10** |
| **Testing** | • Sample output doesn't match the provided expected output<br>• Output is not formatted according to the specification (expected output) | • Parts of the sample output don't exactly match the expected output<br>• Output formatted according to the specification (expected output) | • Sample output exactly matches the provided expected output<br>• Output formatted according to the specification (expected output) | **/10** |
| | | | **Total** | **/100** |