**Air Route Planner**

Software Requirements Specification

**That “One” Team**

**SE 300 Section 01**

Table of Contents

[**Introduction** 3](#_Toc348991000)

[**Team Project Information** 3](#_Toc348991001)

[**Schedule** 3](#_Toc348991002)

[**Requirements Specification** 4](#_Toc348991003)

[**External Interface Requirements** 7](#_Toc348991004)

# **Introduction**

The purpose of this document is to specify the requirements for the development of an Air Route Planner. . The ARP operates only during a 24 hour period (0000-2359).The system will receive airport information through a text file which contains airport route, carrier name, depart airport, arrival airport, arrival and departure times and cost. The system will calculate a travel route based on the following parts: cost, time and airline available for that time.

# **Team Project Information**

Course: Spring 2013 SE300 Section 1

Team: That “One” Team

Members/Roles:

* **Team Leader:** Brian Powell
* **Development Manager:** Brittany Rompa
* **Planning Manager:** Craig Wilkerson
* **Quality Manager:** Yutong Zhu
* **Req/Support Manager:** Muraad Khan

# **Schedule**

Projects launch 01/30/2013

Github & Requirements questions 02/01/2013

Strategy/Plan 02/04/2013

Project plan 02/06/2013

Interview 02/08/2013

Task 02/10/2013

Requirements 02/11/2013

Requirements 02/13/2013

# **Requirements Specification**

User Interface Requirements

1. The user interface for the program shall be of a graphical nature.
   1. The interface shall allow the user to enter:
      1. Origin
      2. Destination
      3. Sorting Criteria
         1. Cost
         2. Time
         3. Airline
         4. Airport
            1. Service to
            2. Service from

* 1. The interface shall display the results to the user
  2. The interface shall allow the user to:
     1. Add/Modify airports
     2. Add/Modify routes
        1. Airlines based on existing routes
     3. Delete
        1. Airports
        2. Routes
     4. Display most current route information (i.e. all routes in network)
     5. Display airports that can be reached from a given airport
     6. Display air route information for a given route
     7. Close airport
        1. Re-open closed airport
     8. Save any modifications made

Data Input Specifications

1. Data will be read in from a text file
   1. Data will be separated into sections denoted by ‘#’
   2. The keyword “#comment” will designate a comment line that is not to be parsed by the system
      1. Must be first word in commented line
   3. No upper bound to the number of airports, routes or closures listed in the file
   4. All data fields are of type string
   5. Optional fields are allowed
      1. If an optional field is used, the system must be able to parse data that does not contain values in these fields
      2. If an optional field is not used, the system must be able to parse data that is contained in these fields
   6. The system must recognize bad data
      1. System will alert the user that bad data was encountered
      2. The data must be either discarded or tagged as unusable
   7. Once the data is loaded, there are no restrictions on ordering of data or data type

Data Modification Requirements

1. System will allow user to enter new routes
   1. New routes will require the following information
      1. Airline
      2. Departure airport
         1. Error if the airport is not already in the network
      3. Departure time
         1. Error if the departs before midnight
      4. Arrival airport
         1. Error if the airport is not already in the network
      5. Arrival time
         1. Error if the flight time is less than thirty minutes
         2. Error if the flight arrives before it departs
         3. Error if the flight arrives after midnight
      6. Price
         1. Error if the price is less than zero
2. System will allow user to add new airports
   1. Airports must have a three letter call code
   2. Airports must be initialized with no added routes
3. System will allow user to delete routes
4. System will allow user to delete airports
   1. Any routes associated with deleted airport must be deleted as well
5. System will allow user to close airport
   1. Airport must remain closed until user re-opens airport
   2. Error must be generated if user attempts to close a closed airport
6. System shall only save modifications at user’s request
   1. If requested, all data will be written to the input file
      1. Additions will be appended to the end of the file
      2. Deletions will remove the data from the file
   2. Any modifications will automatically appear in other related fields

System Output Requirements

1. System will display results based on filter selected
   1. Cheapest route filter will return the following information:
      1. Total price of flight
      2. Total travel time (including layover)
      3. Names of paths used
   2. Shortest total travel time filter will return the following information:
      1. Total price of flight
      2. Total travel time (including layover)
      3. Names of paths used
   3. Using a specific airline for the greatest percentage of time will return the following information:
      1. Total price of flight
      2. Total travel time (including layover)
      3. Names of paths used
   4. There will be a filter that will allow the user to display all three filters at once
2. System shall allow the user to see all airports currently in the network
   1. Information must be tabularized for easy viewing
3. System shall allow the user to see all airlines that service a given airport
   1. Information must be tabularized for easy viewing
4. System shall allow the user to see the air route information for all routes
   1. Information must be tabularized for easy viewing
   2. Information must consist of:
      1. Carrier
      2. Departure airport
      3. Departure time
      4. Arrival airport
      5. Arrival time
      6. Price of flight

Non-Function Requirements

1. The program shall use twenty-four hour time format
2. The program shall use appropriate exception handling so that the system responds with a clear, descriptive message when an error or exceptional condition occurs.
3. The implementation programming language must be a “standard” version of a widely used language.
4. The system must be easily portable to a variety of computer environments
5. The system must be easy to maintain

# **External Interface Requirements**

