

Concordia University ENCS

**2009**

**COEN 345 Airline Reservation System Version 2.0**

Written by: Jaime Yu (5428998), Joseph Khater (5077370), Gaurav Nayak (5836883)

**Concordia University Engineering and Computer Science**

Contents

[**Application Requirements** 3](#_Toc246763819)

[**R1: Book a passenger** 3](#_Toc246763820)

[**R1.1: Passport number of passenger** 3](#_Toc246763821)

[**R1.2: Name of passenger** 3](#_Toc246763822)

[**R1.3: Address of passenger** 3](#_Toc246763823)

[**R1.4: Telephone number of passenger** 3](#_Toc246763824)

[**R2: Update booking of a passenger** 3](#_Toc246763825)

[**R2.1: Passport number of passenger** 3](#_Toc246763826)

[**R2.2: Name of passenger** 3](#_Toc246763827)

[**R2.3: Address of passenger** 3](#_Toc246763828)

[**R2.4: Telephone number of passenger** 3](#_Toc246763829)

[**R3: Cancel booking of a passenger** 3](#_Toc246763830)

[**R4: List bookings of a flight** 3](#_Toc246763831)

[**R5: Query booking information** 3](#_Toc246763832)

[**R6: Book a seat for each passenger** 3](#_Toc246763833)

[**R7: Seating arrangement of a plane** 3](#_Toc246763834)

[**Assumptions** 4](#_Toc246763835)

[**Limitation** 4](#_Toc246763836)

[**User Manual** 4](#_Toc246763837)

[**References** 5](#_Toc246763838)

**Application Requirements**

**R1: Book a passenger**

The airline agent will be able to book a passenger to a flight. When the booking will be created it will have the following sub-requirements:

**R1.1: Passport number of passenger**

Each passenger has a unique passport number which starts with a letter and is followed by 10 digits.

**R1.2: Name of passenger**

Each passenger has a name which can be a maximum of 50 characters.

**R1.3: Address of passenger**

Each passenger has an address.

**R1.4: Telephone number of passenger**

Each passenger has a 10 digit long telephone number.

**R2: Update booking of a passenger**

The airline agent will be able to update the booking of a passenger. While updating a booking the agent will be able to update the following fields:

**R2.1: Passport number of passenger**

Each passenger has a unique passport number which starts with a letter and is followed by 10 digits.

**R2.2: Name of passenger**

Each passenger has a name which can be a maximum of 50 characters.

**R2.3: Address of passenger**

Each passenger has an address.

**R2.4: Telephone number of passenger**

Each passenger has a 10 digit long telephone number.

**R3: Cancel booking of a passenger**

The airline agent will be able to delete/cancel the booking of a passenger.

**R4: List bookings of a flight**

The airline agent will be able to list all the bookings made on a particular flight.

**R5: Query booking information**

The airline agent will be able to query the booking information of a particular passenger.

**R6: Book a seat for each passenger**

One seat is assigned to each passenger.

**R7: Seating arrangement of a plane**

All planes contain 20 rows of seats. Each row will have 4 seats identified by using A, B, C and D.

**Assumptions**

1. All flights use the same type of planes.
2. A passenger can only take direct flights (e.g. from city A to B) and not connecting flights (e.g. from city A to C and then from C to B).
3. A passenger can reserve more than one seat on a plane. Therefore, there can be identical passenger information in a flight. Was originally designed for wealthy playboys and obese people who wish to buy more than one seat for more added comfort.
4. Users can create post dated flights and bookings. This has no real value in the real world except to fix clerical mistakes.

**Limitation**

1. There is only one booking database for every flight object. That means that repeating flights do not have bookings for each individual distinct flight. Code was originally designed to create dynamic instances of booking databases for each distinct flight but errors could not be fixed before deadline.

**User Manual**

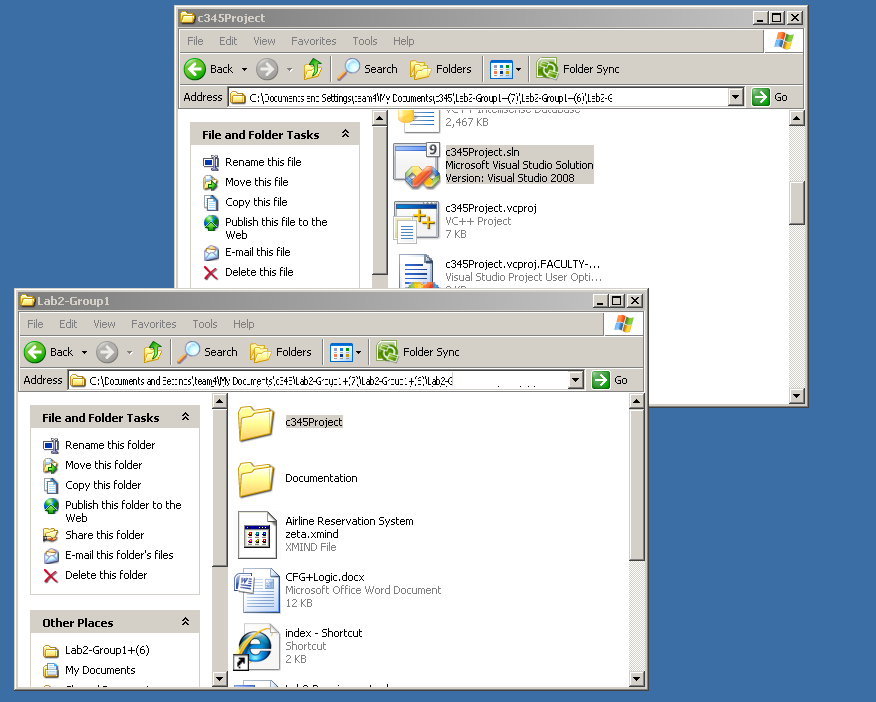
**IMPORTANT INSTRUCTION! **

Please see attached video files for a video tutorial on how to use the provided Airline Reservation System. YouTube versions are located at: <http://www.youtube.com/watch?v=lYhzaN9uApU> or unzip Videos.7z to see the embedded videos. You may also use “Click to Read Documentation.html” for the code documentation of the application along with comments and description of the functions. You can navigate the code much faster using the HTML version because you can click on functions to see their definitions. There is a supplied PDF, OBJECT-DIAGRAM.pdf, with a diagram showing how the application’s objects are related to each other. To help with CFC graphs, the team provided a document called CFG+Logic.docx.

**Loading Source Code**

The code can be used on any version of Microsoft Visual Studio. The team recommends using the latest one: Microsoft Visual Studio 2008 (or Microsoft Visual C++ 2008 Express Edition) to open the solution since it contains the latest updates and it has many new neat features that allow the testers to save time while performing white-box testing.

Below is picture showing the solution file (.sln) that must be double-clicked to open the application in Microsoft Visual Studio.



For further information please contact us from: [http://groups.google.ca/group/**coen-345**](http://groups.google.ca/group/coen-345) or jaime@jaimeyu.com.

**References**

http://upload.wikimedia.org/wikipedia/en/1/1e/Munich\_airport\_flightboard-2.jpg

http://hashlib2plus.sourceforge.net/

http://www.cplusplus.com/