Airline Reservation System (Windows Application)

1. Purpose of Application

The application developed for this assignment is airline reservation system. This airline reservation system can be used by normal clients for booking flights, and also can be used by system administrator to manage amounts of flights, airlines and various seat configurations.

Moreover, this system is also one of the best examples to demonstrate characteristics of distributed system. In my project, I mainly demonstrate security features by using login form and password decryption, and also demonstrate the fault tolerance in case of database service go offline.

2. System Architecture

2.1 Main Components

In my airline reservation system, the necessary component required to run the system should be the database system, database service, and a server host. These three components mentioned above are made up of the server-side service. On the other hand, as for the client-side representation, I apply WPF windows as the front-side presentation for the clients. Details about these components are shown below:

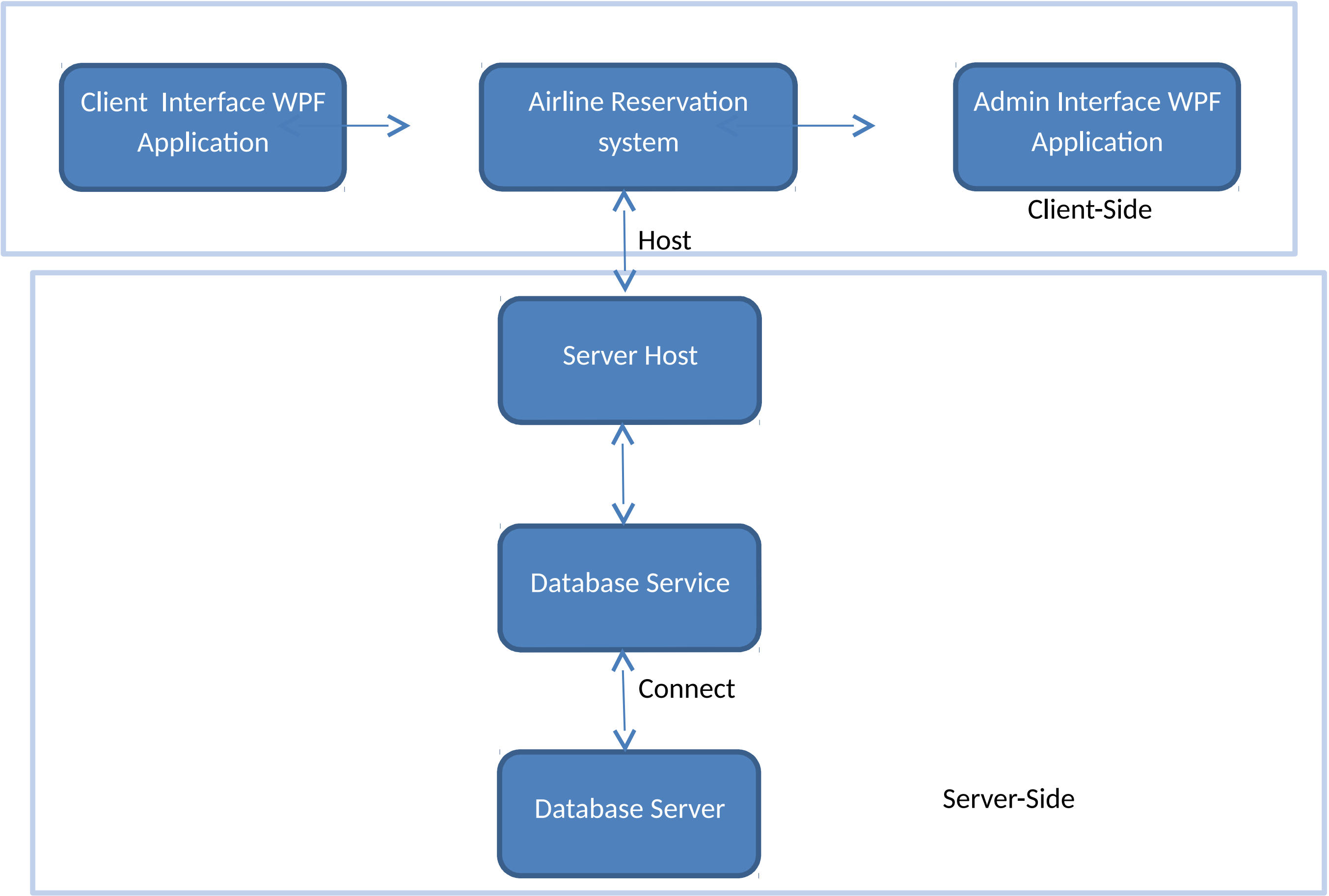
- **Database System:** SQL server is used as the underlying database system, which contains the tables (users, airlines, flights, seats, orders table) and views (viewflights and vieworders).
- **Database Service:** This service is defined in WCF, which including the library and actual service application. Moreover, this service is also divided into two parts (like database service and database backup service) for ease of failure handling. In addition, class diagram and business logic (like sql CRUD operation and other functions) are also implemented in this service node.
- **Server Host:** This host manages all the server-side service designed in my project. What is more, it bind the WCF service to the proper address, so that the client can connect to the server and perform necessary tasks.
- **WPF Client:** These are multiple WPF windows which can navigate with each other, which can be used by client as well as the administrator.

2.2 Distributed Components

The distributed Components of my system should be the WCF service part and the WPF Client. The reason is that WCF service can be bind to any server host available on the web, which makes distributed features available.

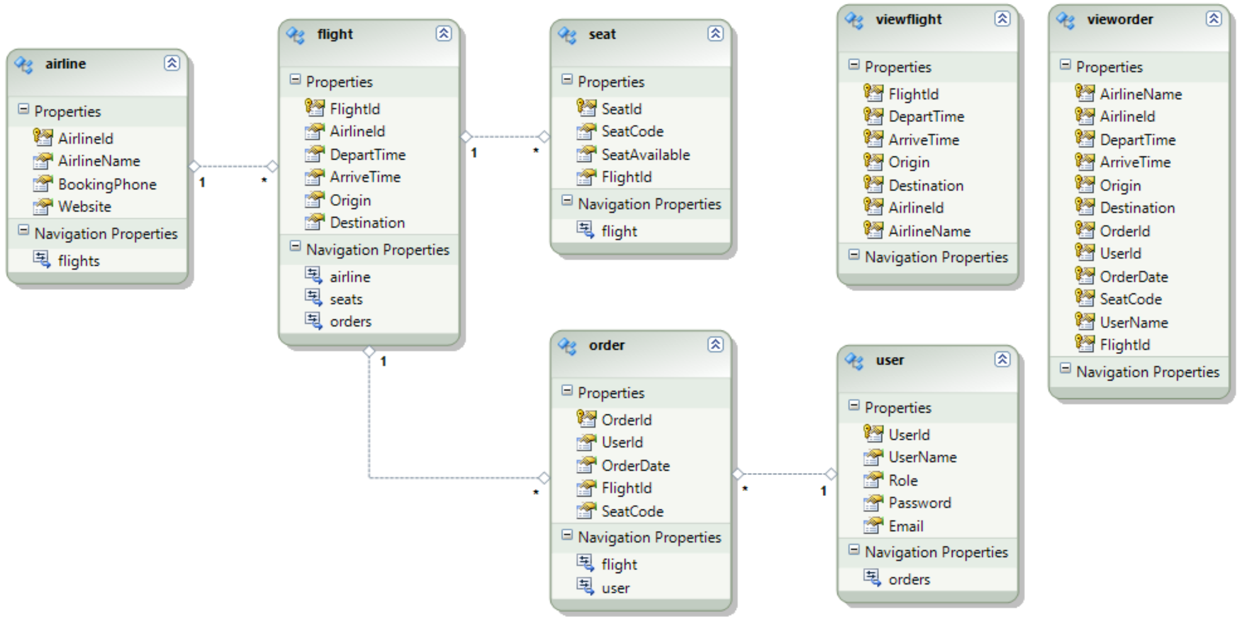
As for the WPF Client, it can be distributed component because the client can get access to any service available on the web by referring to the right service reference. The only thing need to do is to input the right endpoint address.

2.3 System Components Diagram


Client Interface WPF Application
Airline Reservation system
Admin Interface WPF Application
Client-Side
Host
Server Host
Database Service
Connect
Database Server
Server-Side

3. Class Diagrams

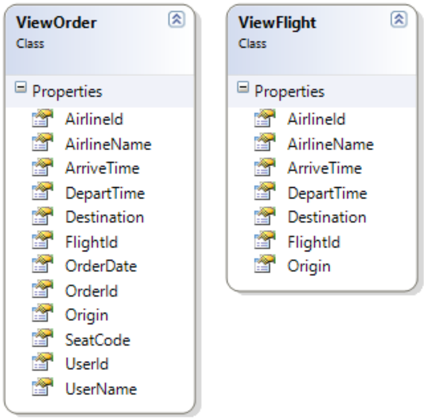
3.1 Database Schemas



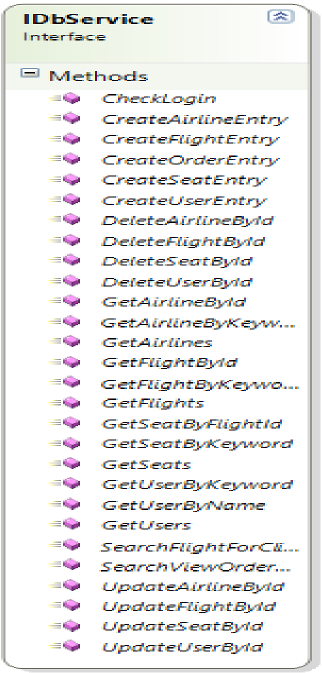
3.2 Classes

Classes Defined For Database Table Object

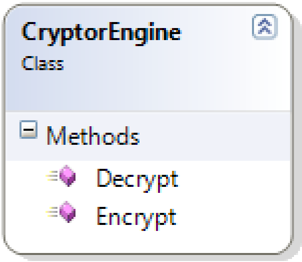
Class Defined For Database View Object



Class Defined For Service Interface (implemented by wcf service)



Other Class for Password Encrypt and Decrypt



4. Known Issues

When the system grows larger, potential issues that may occur are:

- System Latency: Take the login function of the system for example. As the system does not have concurrency function, so the user cannot login concurrently. In order to connect to the system, the user needs to wait until other user finishes the connection.
- System Overload: if there are too many clients connecting to the server, the workload will be too high, which may result in a large degree of service degradation. Even worse, the server may breakdown finally.
- Compatibility Issue: Because the application is developed in the windows development environment, some clients who use Linux, Mac may meet troubles when starting the project.

Bugs:

- Bugs related to update table: Take the airline table in database for example. Because airline Id is selected as the primary key, it is not permitted to update the airline Id by the administrator.
- Bugs related to create record: Take the user table in database for example. Because user Id is selected as the primary key, the user name may be repeated which is not consistent real scenario.
- Bugs related to delete record: Take the airline table and flight table for example. Because the airline id in flight table needs to reference the airline id in airline id, it will cause error if the user wants to delete the airline record which is referenced by flight table.
- Bugs related to user role: It is necessary to build a system which need to verify whether the user has right to register as Admin.