Rationale for Project

The purpose of this project is to design a travel agency software system to manage flight information for customers. Both customers and staffs have access to flight information. However, only staffs have the authority to make any changes to flights upon customers’ requests. Using this system, customers can also earn reward points based on their expenses. During the project requirement discussion, one particular problem that came up is that how much authorization staffs should have towards customer's’ account, and the decision was to give staffs access only to the flight information such as booking or comparing prices with multiple airlines. Whereas, customers will have access to the flight information, rewards, and payment options. So, the main purpose of this project is to develop a software system where customers will be able to access their flight information on the go and staffs will be able to make any changes to the booking as well as personal errors.

Based on the problem statement, several use cases have been designed for the development of the system. These use cases have been categorized based on staffs, customers, and guests. To log in to the system, staffs need to use case 9 (staff login), and staffs are not allowed to change their password. Use case 1 (staff Detail) allows the staffs to check their personal information once they are logged in to the system. Staffs can also access flight information to keep themselves updated of any changes by implementing use case 2 (staff check flight information). It can also be used to check on flight information based on customer requests. Use case 3 (staff check customer information) allows staffs to check customers/clients information such as their names and date of birth to make sure that there is no mistake and everything is up-to-date. If staffs find any mistake, use case 4 (change customer information) is implied to fix that particular mistake upon that customer’s request. Use case 5 (check messages) allows staffs to receive notifications of any changes in flights such as a flight being sold out or the flight departure or arrival time has changed. Once the staffs are aware of these changes they can inform the customers about it.

For customers, the use case 7 (customer and guest login) is used to log in to the system. Once the customer is logged in to the system, he/she has the option to search for flights, access the reward points, access previous or upcoming trips, and last but not the least he/she can recommend friends to sign up for an account. To search flights, use case 11 (search flight) can be utilized by customers. Customers must enter where they are traveling from and the destination plus the departure date and time. Once these information are put in the system, customer can click search and the database gets updated. Once the system finds matching flights based on all the information provided the customer, it will be displayed on the screen. If any information provided by the customer does not match the system database then it will display an error message. Based on the error message, the customer can fix whatever mistake they made while putting their information and search again. Once a customer found a flight schedule that fits his/her availability, use case 12 (Existing customer booking) is used by the customer to book the flight. If the customer is satisfied with the result he/she can continue with checkout by using use case 13 (customer check out). Customers also earn points per trips and they can apply towards their future trips which can be processed using use case 21 (redeem gifts). Using use case 28 (friends recommending), existing users have the option to recommend friends to sign up. That also earns them points which add up to their total points. The recommended friends can also earn some points once they sign up. Use case 20 (change password), gives the option to customers to change their password in the system in timely basis. But if a customer forgets his/her password, they can also retrieve it by using use case 10 (retrieve password).

New guests are not required to sign up for an account. If any guests would like to sign up, use case 35 (guest signup) can be used. He/she will be asked his/her personal information such as name, date of birth, email to open up an account. Without an account, a guest can still search for flights (after logging in through use case 7) in the system the same way a registered customer would (Use Case 11). Use case 16 (guest booking) is created especially for guests to book their flights in the system. Guest do not get points for their trips since none of their inputs gets saved in the system.

These use cases have been combined and put into several categories. Based on that, five classes are created. They are staff, customer, GUI, acctinfo, and flightInfo. The staff class can access staff info using the acctInfo class. Using GUI class, a staff can login to the system. Once he/she is logged in to the system, staff can access flight information using the flightInfo class.

The customer class can either login or signup using GUI class. Once he/she is logged in, the customer can access account information using acctInfo class. He/she can also has the ability to access flight information using flightInfo class. Inside the customer class, a method for redeeming points is also implemented. The system can compute total redeem points for the customer. The acctInfo class has the information of both staffs and customers. Customer class and staff class extend acctInfo class to get personal information. Additionally, staffs can modify customer’s information based on request. The flightInfo class contains all the airlines information that are accessed through database. Both customer and staff has access to flight information thorough acctInfo class. The GUI category contains login GUI for both customer and staff. It also has signup and checkout GUI for customer.

Upon the previous development, the software architecture used for the project is MVC architecture, which is model-view-controller architecture. The model component of this architecture is database. In our project, the model contains airlines, flight information and account information. This lead to the view component of the architecture. View component displays information from the database to the user interface. Lastly, controller component acts as intermediary between view and model component. End-User send request through View with controller, and manipulates the database (model).Once the information is processed the result is sent back to the view for end-users.

Based on the coding for the software development that are completed so far, there were some testing done to some of the methods to check if it’s performing the way it is intended to perform. During the testing, the white-box testing type is used. There are five test cases used to test the T2P software.

Signup test case is used during the signup process for new customer. This particular test case was created while testing the signup process. During the signup process, the new user has to provide required information to the system. The reason for this particular test is that it is crucial for new customer to add into the system, and also avoid the duplicates for security issues.

Login test case is used during the login process for staff and customer. This test-case validates the username and password for staff and customer. Security issue also emerge to the surface. Customer should not have the access to the staff control system. Customer and staff have distinguish database and frame in order to solve the problem.

Searching flight for staff test case is used when staff logged in to the system, and search (match) information in the database base on the given criteria. The test case filters the entire database with given parameters. The database shall return the correct flight information.

Checking flight status for customer test case is used when customer wants to check their specific flight status. It is similar to the searching flight for staff, but this control is provided for customer. The reason for this test case is to validate the correct information for specific flight in the database with additional return factor which is flight status ( flight arrived, on the way, or not yet).

Redeem as customer test case is used when a customer wants to check or redeem his/her gift points. This test case checks how methods interact with database. The method calculates the correct gift points available for customer. It also validates if his/her gift points are redeemable ( not enough points). The test case checks if the software gives notification for the customers when they redeem.