GUI design phase

Planned time: 24/04/2015 – 08/05/2015

Actual time: 24/04/2015 – 08/05/2015

Actual Work:

Started by creating a mockup of the systems front-end, this included the front page, home pages for each user and example results, search and data-entry pages. A few days later a complete rework and revamp from scratch was done to make a much more impressive look to show the clients. Initial design made using HTML5, CSS3 and JQuery 1.8. This was again changed, altering the layout and structure of the site. Navigation now comes through a central iframe with all subsystems and their various options in sub-menus. This would later require the back-end to navigate through the various functionalities as well as included font-awesome for glyphicons to add to overall feel of website.

New element:

* Admin.html
* Agent.html
* Bg.jpg
* Bg01.jpg
* Fms.css
* Fmanager.html
* Index.html
* Functions.js
* Smanager.html
* Staff.html
* User.html
* Font-awesome libraries
* JQuery libraries
* Canvasjs libraries

Database design and implementation phase

Planned time: 24/04/2015 – 08/05/2015

Actual time: 24/04/2015 – 12/05/2015

Actual Work:

At the beginning, we designed the ER diagram for the database. After consulting with the client several times, we did some modification to the diagram and started constructing the actual database. We constructed the database using MySQL.

New element:

* Create MySQL quires, tested by Siyuan Hou
* Create Customer Class and add the methods to communicate with the database, tested by Siyuan Hou
* Create Airport Class and add the methods to communicate with the database, tested by Siyuan Hou
* Create Agent Class and add the methods to communicate with the database, tested by Siyuan Hou
* Create SeatMap Class and add the methods to communicate with the database, tested by Siyuan Hou
* Create Route Class and add the methods to communicate with the database, tested by Siyuan Hou
* Create Ticket Class and add the methods to communicate with the database, tested by Siyuan Hou
* Create Schedule Class and add the methods to communicate with the database, tested by Siyuan Hou
* Create Fleet Class and add the methods to communicate with the database, tested by Siyuan Hou
* Create ServiceInventory Class and add the methods to communicate with the database, tested by Siyuan Hou
* Create Service Class and add the methods to communicate with the database, tested by Siyuan Hou
* Create User Class and add the methods to communicate with the database, tested by Siyuan Hou
* Create UserAuthority enumeration Class and add the methods to communicate with the database, tested by Siyuan Hou

Subsystem methods design and implementation phase

Planned time: 08/05/2015 – 15/05/2015

Actual time: 12/05/2015 – 17/05/2015

Actual Work:

During this phase, the objective is to write functions for subsystems. At first, we reviewed our system requirements about subsystems. Then we referred to the entity database and wrote down the functions necessary to implement. In addition to that, we also reviewed the GUI we implemented earlier because we need each component in GUI compatible with the database. Also, we needed redesign some of our entities to make them appropriate.

New Elements:

* Create MySQL quires, tested by Siyuan Hou, Sandon Joubert
* Create ProfileSystem Class and add the methods to interact with GUI and entity classes, tested by Siyuan Hou
* Create ReservationSystem Class and add the methods to interact with GUI and entity classes, tested by Siyuan Hou
* Create ServiceSystem Class and add the method to interact with GUI and entity classes, tested by Siyuan Hou
* Create ReportSystem Class and add the method to interact with GUI and entity classes, tested by Siyuan Hou
* Create class AgentReport (a subclass of ReportSystem Class) and add the method to interact with GUI and entity classes, tested by Siyuan Hou
* Create class PassengerReport (a subclass of ReportSystem Class) and add the method to interact with GUI and entity classes, tested by Siyuan Hou
* Create class MonthlyBookingReport (a subclass of ReportSystem Class) and add the method to interact with GUI and entity classes, tested by Siyuan Hou
* Create class CustomerReport (a subclass of ReportSystem Class) and add the method to interact with GUI and entity classes, tested by Siyuan Hou
* Create class CashierReport (a subclass of ReportSystem Class) and add the method to interact with GUI and entity classes, tested by Siyuan Hou
* Create class DailyBookingReport (a subclass of ReportSystem Class) and add the method to interact with GUI and entity classes, tested by Siyuan Hou

Dao Classes methods design and implementation phase

Planned time: 17/05/2015 – 28/05/2015

Actual time: 17/05/2015 – 29/05/2015

Actual work:

During this phase, the objective is to implement the whole system. Based on what is implemented previously, we wrote down the functions needed to get entity data and transmit between subsystems and GUI. In a word, these functions act as connection between entities and subsystems. We also had to redesign some of our subsystem functions.

Merge of back-end and front-end

A lot of getting the back-end to work with front-end was day overnight and only need some small changes / additions or IDs, Names and Value attributes for it to properly work with the back-end. Had to separate many of the functionalities into separate files and then remember that JS and CSS doesn’t pass over through iframes and I had to add links to JS and CSS in all of them. Lastly just added classes to some forms and tables which did not have them.

New Elements:

* Create MySQL quires, tested by Siyuan Hou, Sandon Joubert
* Create class CustomerDao and add methods to interact with both subsystems and entities databases.
* Create class AirportDao and add methods to interact with both subsystems and entities databases.
* Create class AgentDao and add methods to interact with both subsystems and entities databases.
* Create class SeatMapDao and add methods to interact with both subsystems and entities databases.
* Create class RouteDao and add methods to interact with both subsystems and entities databases.
* Create class TicketDao and add methods to interact with both subsystems and entities databases.
* Create class ScheduleDao and add methods to interact with both subsystems and entities databases.
* Create class FleetDao and add methods to interact with both subsystems and entities databases.
* Create class ServiceInventoryDao and add methods to interact with both subsystems and entities databases.
* Create class ServiceDao and add methods to interact with both subsystems and entities databases.
* Create class UserDao and add methods to interact with both subsystems and entities databases.
* addScheduleForm.jsp
* addServiceInventoryForm.jsp
* addStaffForm.jsp
* error.jsp
* errorWithoutLogin.jsp
* getCustomerEmailForm.jsp
* homeAgent.jsp
* homeCustomer.jsp
* homeStaff.jps
* index.jsp
* login.jsp
* modifySchedule.jsp
* modifyStaff.jsp
* register.jsp
* searchScheduleForm.jsp
* searchScheduleWithoutLogin.jsp
* showAccountInfo.jsp
* showAgentInfo.jsp
* showAgentsCustomerList.jsp
* showAllMyBooking.jsp
* showAllServiceInventory.jsp
* showBookingDetail.jsp
* showCreditCardInfo.jsp
* showCustomerInfo.jsp
* showCustomerReport.jsp
* showFlightDetails.jsp
* showScheduleSearchResult.jsp
* showSearchServiceResults.jsp
* showServiceDetail.jsp
* success.jsp

Test phase

Planned time: 28/05/2015 – 31/05/2015

Actual time: 28/05/2015 – 31/05/2015

Actual Work:

New elements: