Group5

Flight Management System

CSCI222 System Development

Ruixi He

Zheli Jiang

Junyan Fan

Siyuan Hou

Sandon Joubert

Contents

[**1. Business Case** 3](#_Toc417388480)

[1.1 Overview of the project 3](#_Toc417388481)

[1.2 Stakeholders 3](#_Toc417388482)

[*1.2.1 Who are the stakeholders?* 3](#_Toc417388483)

[*1.2.2 What will this system do for them?* 3](#_Toc417388484)

[1.3 Business Value 4](#_Toc417388485)

[**2. Detailed Plans** 6](#_Toc417388486)

[**3. Risk and Counter Measures** 12](#_Toc417388487)

[**4. Software Requirements Specification** 13](#_Toc417388488)

[Revision History 13](#_Toc417388489)

[4.1 Introduction 14](#_Toc417388490)

[*4.1.1 Purpose* 14](#_Toc417388491)

[*4.1.2 Scope* 14](#_Toc417388492)

[*4.1.3 Definitions, Acronyms, and Abbreviations* 14](#_Toc417388493)

[*4.1.4 References* 14](#_Toc417388494)

[*4.1.5 Overview* 15](#_Toc417388495)

[4.2 Overall Description 16](#_Toc417388496)

[*4.2.1 Product Perspective* 16](#_Toc417388497)

[*4.2.2 Product Functions* 18](#_Toc417388498)

[*4.2.3 User Characteristics* 18](#_Toc417388499)

[*4.2.4 Constraints* 19](#_Toc417388500)

[*4.2.5 Assumptions and Dependencies* 19](#_Toc417388501)

[4.3 Specific Requirements 20](#_Toc417388502)

[*4.3.2 Non-functional Requirements* 27](#_Toc417388503)

[**5. Use Cases** 29](#_Toc417388504)

[5.1 Admin 29](#_Toc417388505)

[5.2 Manager 32](#_Toc417388506)

[*5.2.1 Flight Manager* 34](#_Toc417388507)

[*5.2.2 Service Manager* 36](#_Toc417388508)

[5.3 Staff 38](#_Toc417388509)

[5.4 General Public 41](#_Toc417388510)

[**6. Domain Model** 45](#_Toc417388511)

[6.1 Class Diagram 45](#_Toc417388512)

[6.2 Detailed Description 47](#_Toc417388513)

[**7. Meta-report** 60](#_Toc417388514)

[7.1 Tabular Summary of the Group Structure 60](#_Toc417388515)

[7.2 Group Meeting Summary 61](#_Toc417388516)

[*7.2.1 Report of Group Meeting1* 61](#_Toc417388517)

[*7.2.2 Report of Group Meeting2* 63](#_Toc417388518)

[*7.2.3 Report of Group Meeting3* 65](#_Toc417388519)

[*7.2.4 Report of Group Meeting4* 67](#_Toc417388520)

[*7.2.5 Report of Group Meeting5* 69](#_Toc417388521)

[7.3 Individual Work Diaries 70](#_Toc417388522)

[*7.3.1 Ruixi He:* 70](#_Toc417388523)

[*7.3.2 Zheli Jiang:* 75](#_Toc417388524)

[*7.3.3 Junyan Fan:* 77](#_Toc417388525)

[*7.3.4 Siyuan Hou:* 79](#_Toc417388526)

[*7.3.5 Sandon Joubert:* 81](#_Toc417388527)

[7.4 Screenshot of Version Control Software (GitHub) 83](#_Toc417388528)

[8. Member Contribution Assessment 84](#_Toc417388529)

[**Appendix** 85](#_Toc417388530)

**1. Business Case**

1.1 Overview of the project

Our project is a flight management system that provides flight services of a major airline. This system is divided into several subsystems that will automate major operations of the airline. Users are divided into several user groups with different privileges.

1.2 Stakeholders

*1.2.1 Who are the stakeholders?*

System Administrator

Manager

Staff

General Public

*1.2.2 What will this system do for them?*

For administrators, our system allows them to manage personnel and they have the greatest power:

* Create user groups
* Promote staff
* Everything a manager or a staff can do

For managers, our system allows them to manage flights:

* Create flight
* Modify flight
* Order and manage services on plane

For staff, they can do modification to flights:

* Switch seats

For public (customer), different types of customer have different rights:

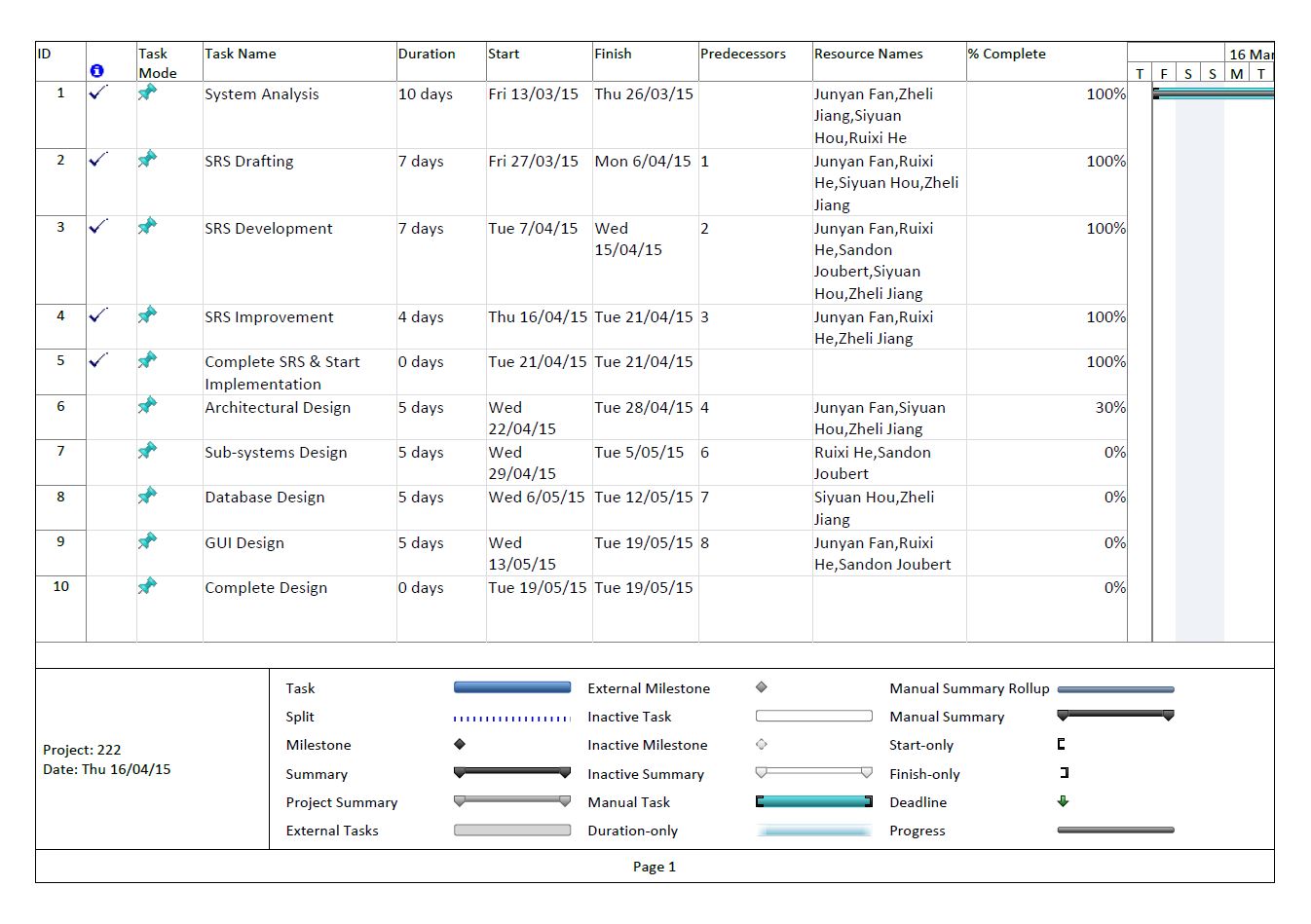
* Agency can have discounts
* Agency can view more specific flight information than regular customers

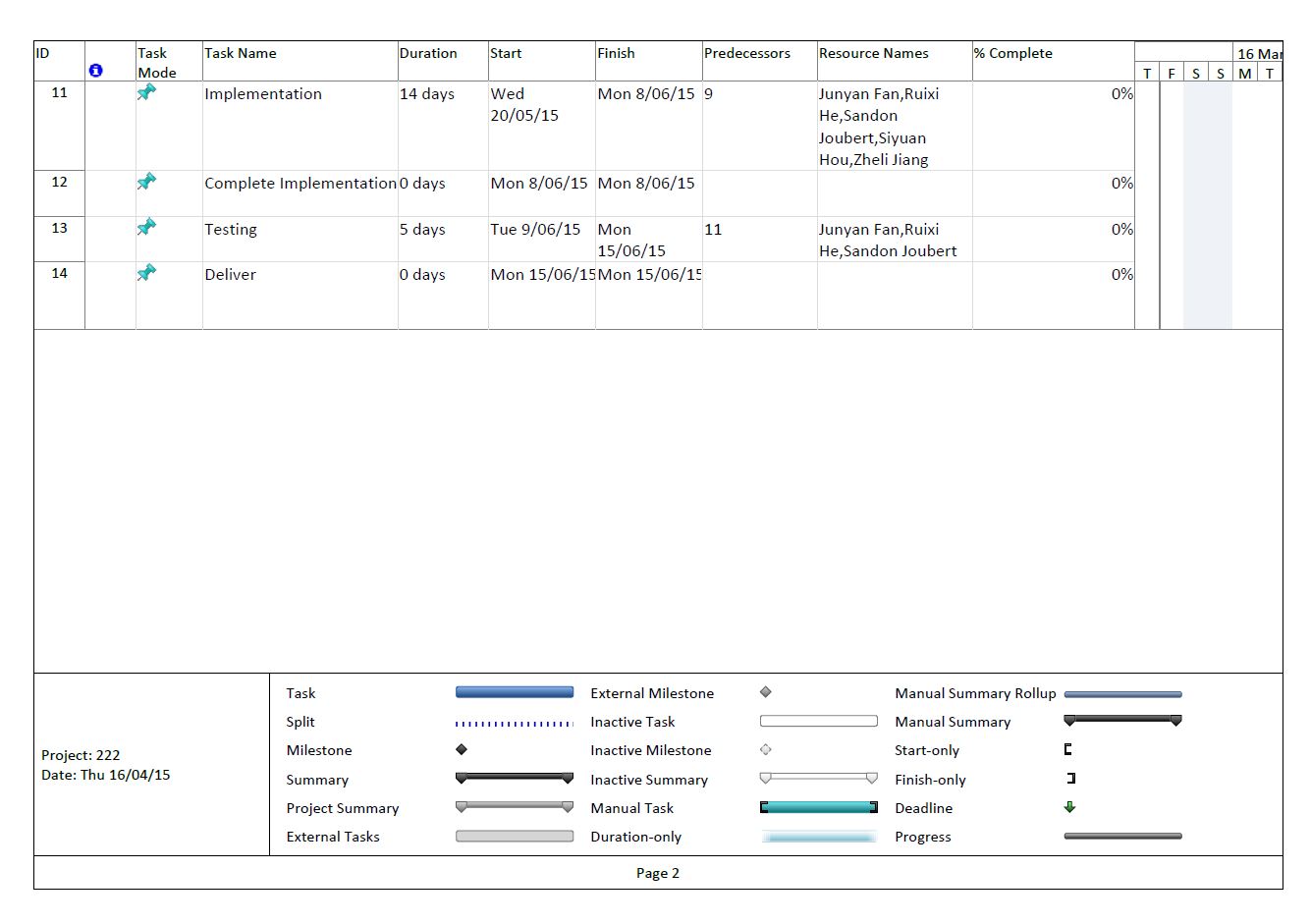
1.3 Business Value

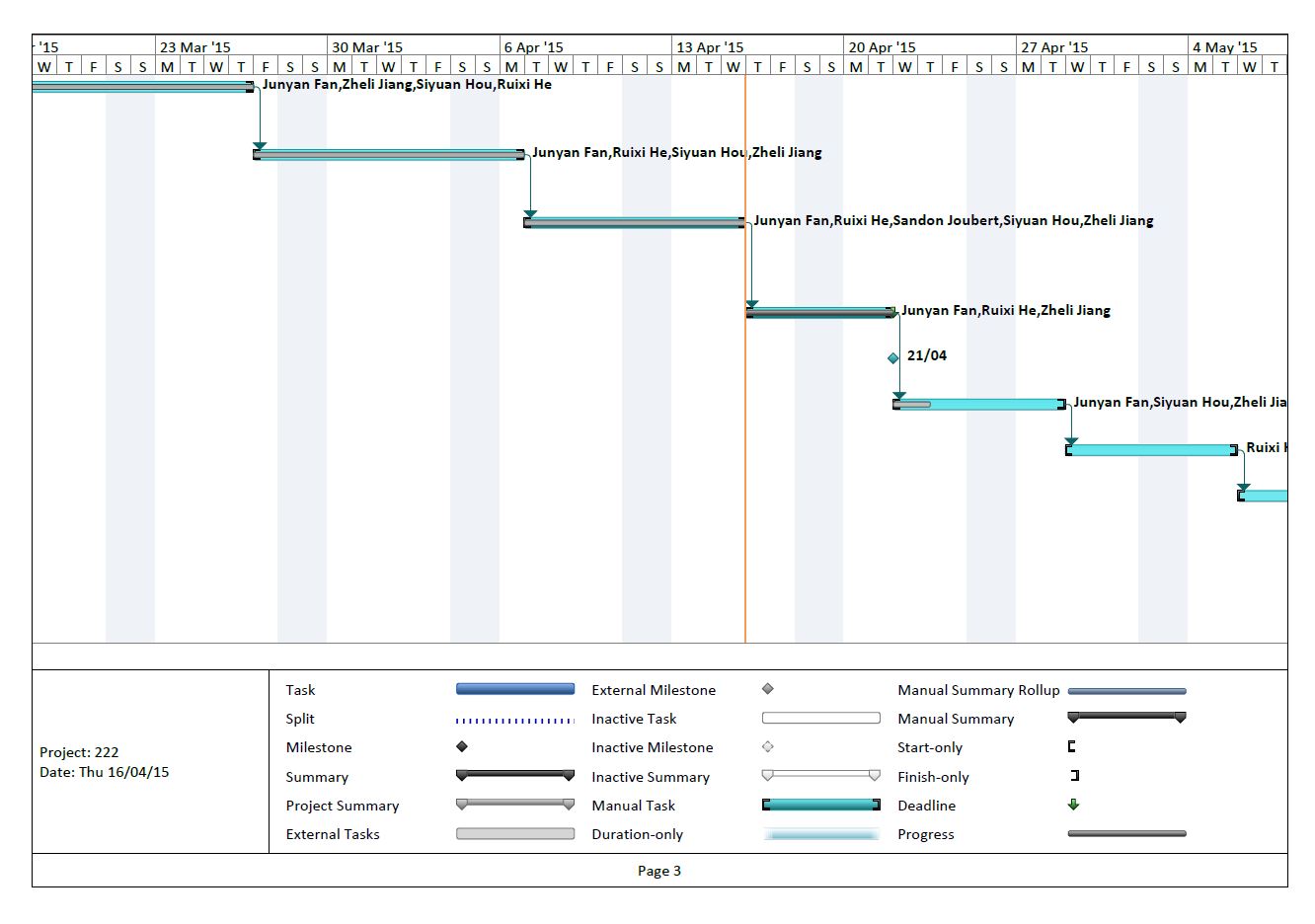
We expect the Flight Management system to reduce labor cost since the current paper-based work requires more personnel. The system is also expected to save time to earn more efficiency. Furthermore, the system should benefit from improved customer satisfaction and increased brand recognition due to its software system.

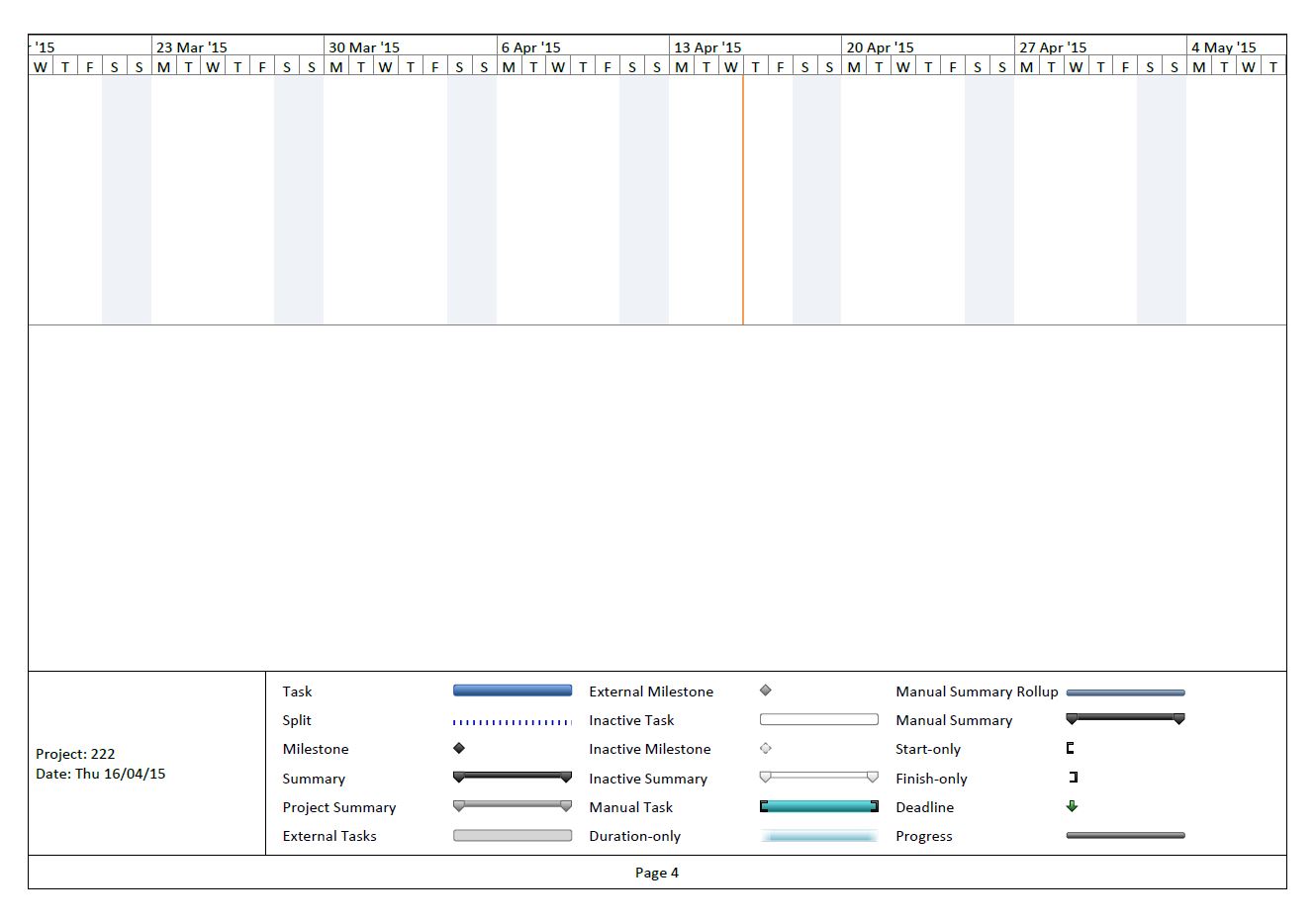
Conservative estimates of tangible value to the company include:

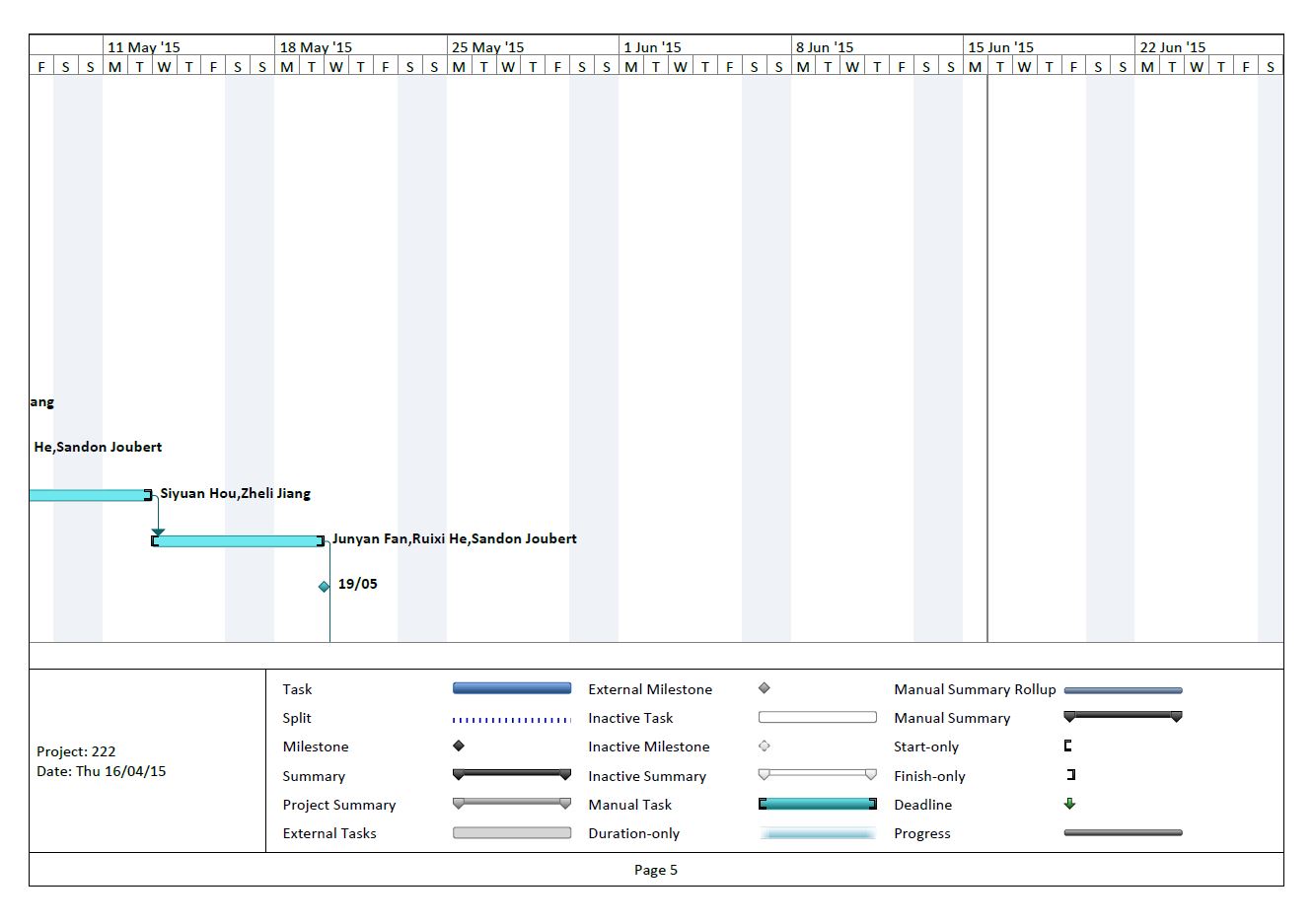
* 30% in sales from new customers
* 40% in sales from existing customers

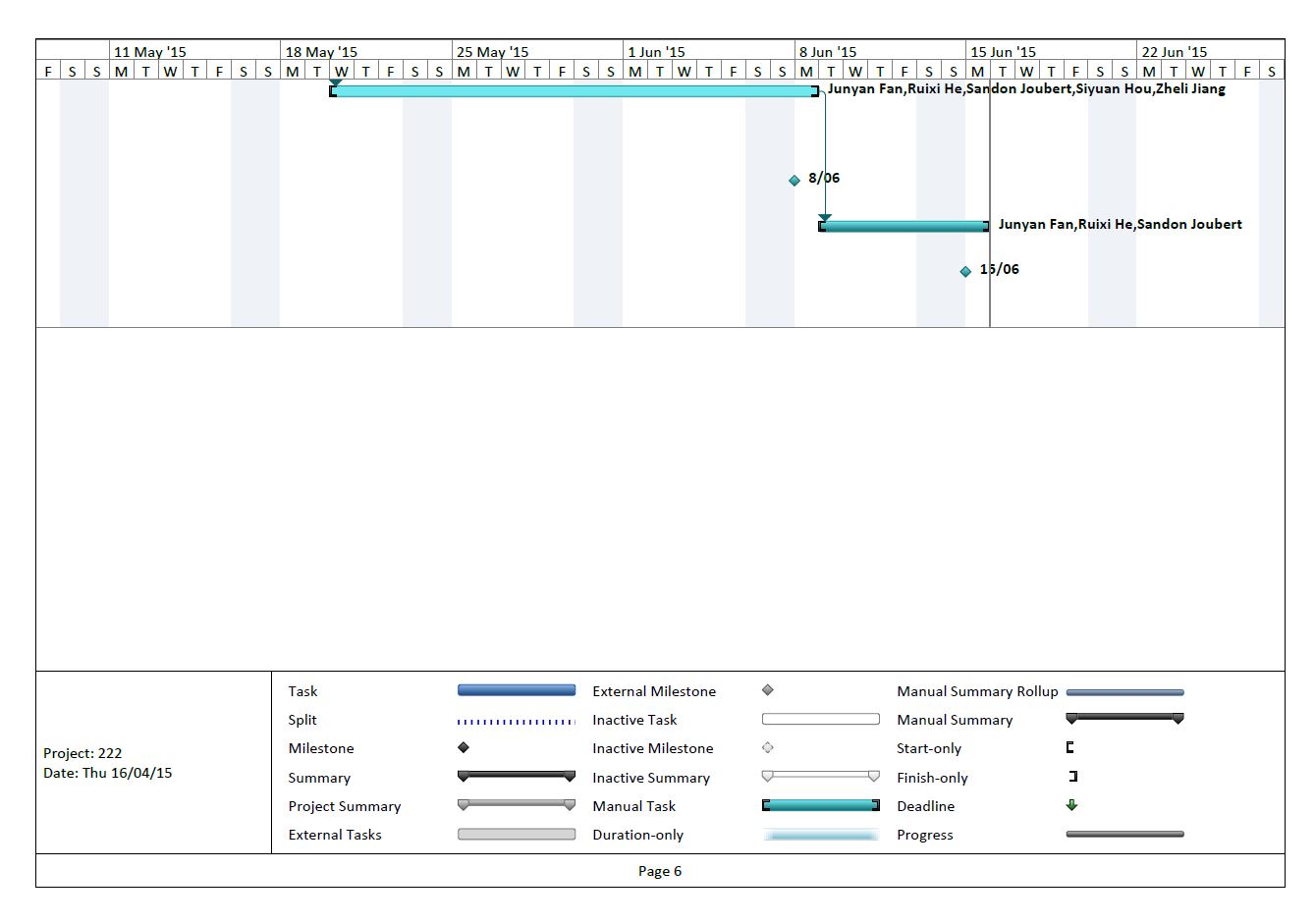
**2. Detailed Plans**











**3. Risk and Counter Measures**

|  |  |
| --- | --- |
| Risk | Counter |
| Using Java programming language for the backend of the Airline Booking System while four members are proficient in Java one has only a little experience with it. This predominately puts the programming workload onto four of the members. | We must assume this risk. Four members should still be enough with the lead programmer still doing a majority of the work. The fifth still knows enough of Java to understand the code. |
| Language barrier can be a problem in communicating. Four members first language is Chinese mandarin / Cantonese whilst the fifths is English. | We must assume this risk with main communication being in English. |
| We only have about 10 weeks to complete this Airline Booking System, with a deadline for a large project like this and all of us having other commitments including other university work and other time consuming tasks outside of university we may struggle to complete the program on time. | By meeting every week and discussing what we will do in the following week as well as keeping diaries for our individual work should hopefully keep us from falling behind. |
| Incomplete or unwritten information and plans. | Avoiding this risk is priority as it will lead to a harder development phase. Using Q & A sessions with client productively will allow up to get all the information we need. |
| Plan carrying out by the team members who have not yet worked in the same or similar projects. | This risk must be assumed as we cannot gain experience on this sort of project over night. |
| Frequent and large change of requirements. | As we continue to ask the clients our understanding of the requirements may change. As long as we can get them finished before the development phase then we will avoid this risk. |
| Unclear requirements. | Speaking to the client can help us avoid and ambiguity in the project requirements. |

**4. Software Requirements Specification**

|  |  |
| --- | --- |
| Flight Management System | Version |
| Software Requirements Specification | **Date: 2.0** |

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Author |
| 20/03/2015 | 1.0 | First SRS | Junyan Fan  Zheli Jiang  Sandon Joubert  Ruixi He |
| 21/03/2015 | 1.1 | |  | | --- | | Form the whole structure of the SRS (Introduction, Overall description, Specific requirements) | | Ruixi He |
| 27/03/2015 | 1.2 | |  | | --- | | Classify the specific requirements into three subsystem (Authentication, Project assignment and Effort estimation subsystem) | | Zheli Jiang  Ruixi He |
| 29/03/2015 | 1.3 | Modify details of specific requirements | Zheli Jiang  Ruixi He |
| 03/04/2015 | 1.4 | Add missing specific requirements related to Adimin | Zheli Jiang  Ruixi He |
| 17/04/2015 | 1.5 | Add non-functional requirements | Zheli Jiang  Ruixi He |
| 21/04/2015 | 2.0 | Finalize SRS | Junyan Fan  Zheli Jiang  Sandon Joubert  Ruixi He |

4.1 Introduction

*4.1.1 Purpose*

The purpose of this document is to describe the specifications on the external behaviors of a flight management system. Besides, it also documents nonfunctional requirements, design constraints and other factors necessary to provide a complete and comprehensive understanding of the system.

The intended audience includes the potential users of the system and software development team.

*4.1.2 Scope*

The software system to be produced is a web-based flight management system.

Our system aims to serve four groups of user:

|  |  |
| --- | --- |
| Roles | Purpose |
| Administrator | Maintain smooth operations of the system |
| Manager | Manage flight resources |
| Staff | Interact with manager and customer |
| General Public | Use the system to book flights |

Our system has four subsystems:

* A Reservation System that manages (e.g. add, change, and modify) all flights reservations, seat selection, ticketing, flight availability, flight details, rates and conditions.
* A Profile Subsystem that manages individual passengers and travel agency profiles.
* A Service Subsystem that manages in-flight services such as food and drinks.
* A Reporting Subsystem to generate various summary report such as: Passengers Report for a day including the occupancy rate etc., various summary Cashier Reports like total revenue for the day, or weeks etc., Monthly Booking Activity Summary, Daily Booking Activity Summary, etc.

Different user groups have different powers. For example, agency is provided with discounts. Also, agency is able to see specific flight information that a normal customer cannot see despite the fact that agency belongs to general public group.

*4.1.3 Definitions, Acronyms, and Abbreviations*

Refer to Appendix.

*4.1.4 References*

* **Rational Unified Process**, SRS template (upedu\_srs.doc), COSC2151 Final Year Software Engineering Project , RMIT International University Vietnam, 2004
* Sample - SoftwareRequirementsSpecification.pdf

*4.1.5 Overview*

The rest of this Software Requirements Specification is divided into two main sections:

* The Overall Description (section 4.2) describes the general factors that affect the system and its requirements.
* The Specific Requirements (section 4.3) contains all software requirements that the system must meet in order to satisfy the needs of customer.

4.2 Overall Description

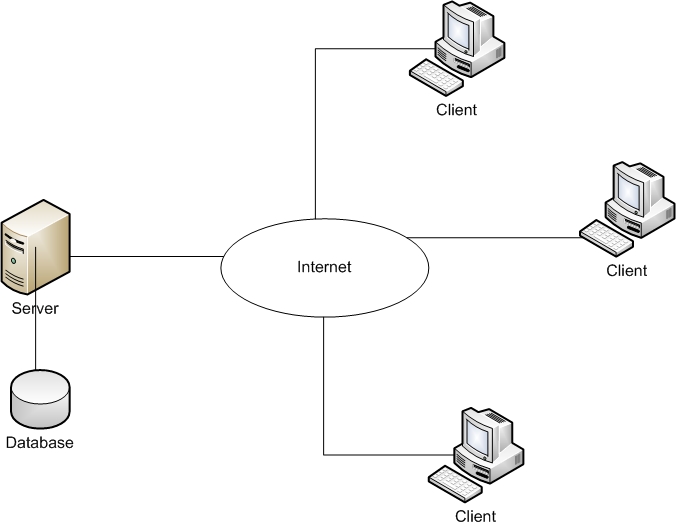
*4.2.1 Product Perspective*

Currently, some of the activities in some flight companies are carried out and managed manually, which makes the process of booking flights difficult to perform and requires an incredible amount of time and efforts.

Therefore, building this system appears necessary and essential. Flight Management System aims to be a perfect tool for achieving goals such as managing flights.

*4.2.1.1 System Interfaces*

The Flight Management System is a web-based application that allows users to interact with the system via internet or intranet.



The clients can simultaneously log into the system from any personal computer that has internet services, and then the clients can interact with the system to do reservation, view profiles, check flight information, etc.

The server takes and processes the clients' request and then retrieves the database and return the result to the clients.

The database is used to store the entire user's profiles, flight information, in-flight services.

*4.2.1.2 User Interfaces*

The user interface provided by the system must be GUI and must be accessible through any web-browser. Depending on the different privileges of the user, they will have slightly different interfaces and the access to slightly different functionalities of the system. For instance, admin has the highest privilege, which means he/she has the right to access all the functionalities provided by the system.

The database should be powered by MySQL server and performed indirectly through the GUIs provided by the system.

*4.2.1.3 Hardware Interfaces*

All components must be able to execute on any computer, which runs on any OS operation system with Java Running Environment.

*4.2.1.4 Software Interfaces*

4.2.1.4.1 User Interface

The user interacts with the system through web browser.

The system supports any web browser.

*4.2.1.5 Communication Interfaces*

The client machines must communicate with the Web Server over TCP/IP connection.

We might have the Web Server and the Database Server are located on different servers.

*4.2.1.6 Memory Constraints*

The client machine must be able to operate within 128MB minimum (including memory for browser)

The Web Server and the Database Server must be able to operate within 512MB minimum.

*4.2.1.7 Operations*

The Flight Management System needs to be easy to use for all users. For customers and agencies, we designed the GUI of this system based on the common sense, which means when the first time the customers or agencies use the system, they will know what 's this button does, and what the next screen will be. And the admin, managers and staff will also be very easy to get used to the functions of the system.

The server installation and maintenance should also be simple for the admin, and we will provide instruction material that teaches the admin how to install and maintain the system.

Backup and recovery must be specified in case of the network failure, database failure, out of power etc.

*4.2.2 Product Functions*

For administrators, our system allows them to manage personnel and they have the greatest power:

* Create user groups
* Promote staff
* Everything a manager or a staff can do

For managers, our system allows them to manage flights:

* Create flight
* Modify flight
* Order and manage services on plane

For staff, they can do modification to flights:

* Switch seats

For public (customer), different types of customer have different rights:

* Agency can have discounts
* Agency can view more specific flight information than regular customers

All functionalities of the system are built on the needs of any flight management system so that this system can make all activities and tasks carried out by users easier and more convenient.

*4.2.3 User Characteristics*

The users of Flight Management System include system administrators, managers, staff and general public.

* Administrators have strong knowledge on networks and web applications to be able to install and maintain the system.
* Managers have good knowledge on web applications and flight information.
* Staff has solid understanding of flights and good interaction skills.
* General public are people who have enough understanding on the use of Internet to use the system.

*4.2.4 Constraints*

The system should strictly obey and satisfy the following constraints:

* Authentication security: the system should enforce user authentication security
* Access control: the system must provide appropriate access right and user interface to each type of user.
* Backup and recovery: the backup and recovery of all the system’s database must be easy to perform to prevent databases from corruption and loss risks
* Integrity control: since the system consists of many databases that are correlated with each other, integrity among these databases must be strictly maintained

*4.2.5 Assumptions and Dependencies*

The following assumptions and dependencies for the system are stated:

* All potential users must have access to Internet.
* All potential users must have a valid email address.

4.3 Specific Requirements

Each requirement (either functional or non-functional requirements) of Flight Management System is ranked based on its level of priority:

* Critical: highest priority level, these requirements are the core functionalities of Flight Management System and must be firstly implemented
* Essential: second priority level, these requirements are the important functionalities of Flight Management System and should be implemented when all the Critical requirements have been finished
* Desirable: medium priority level, these requirements are the necessary functionalities of Flight Management System and should be covered when Critical and Essential requirements have implemented
* Optional: lowest priority level, these requirements are the enhanced functionalities of Flight Management System and should be considered only when all Critical, Essential and Desirable requirements are completed

*4.3.1 Functional Requirements,*

This section includes all the functionalities that Flight Management System provides to System Admin, Flight Manager, Staff, and Customer

*4.3.1.1 Reservation Subsystem*

A Reservation System that manages (e.g. add, change, and modify) all flight reservations, seat selection, ticketing, flight availability, flight details, rates and conditions.

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Requirement type**  Functional | **Use case** |
| **Description:** the system should provide a GUI to let manager input a flight’s detail, like flight time, departure place, arrived place and price | | |
| **Rationale:** flight manger wants to create a new route, so he needs to set all detail | | |
| **Source:** flight manager | | |
| **Fit criterion**: a new rout will be apply | | |
| **Dependencies:** none | | |
| **Rank of importance:** Critical | | |
| **Supporting materials:** None | | |
| **History** create by Junyan Fan 27/03/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Requirement type**  Functional | **Use case** |
| **Description:** the system should provide a GUI to let manager check all routes and manager can select one and change detail about rout, the detail like flight time, departure place, arrived place and price | | |
| **Rationale:** flight manager want to change any rout’s detail | | |
| **Source** flight manager | | |
| **Fit criterion:** new detail will be saved and apply | | |
| **Dependencies**： the data must exist. | | |
| **Rank of importance**：Critical | | |
| **Supporting materials**：None | | |
| **History** create by Junyan Fan 27/03/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Requirement type**  Functional | **Use case** |
| **Description**: system should provide a GUI, which can display all flights’ state, manager can choose one, and display the detail about the plane. | | |
| **Rationale**: managers want to check every plane’s state they can do other operation later. | | |
| **Source**: manager | | |
| **Fit criterion**: all plant’s detail will display. | | |
| **Dependencies**: none | | |
| **Rank of importance**：Essential | | |
| **Supporting materials**：None | | |
| **History** create by Junyan Fan 27/03/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** This system should provide a Staff with a GUI to search flight information. | | |
| **Rationale:** A staff wants to search for flight information. | | |
| **Source:** Staff | | |
| **Fit Criterion:** A GUI form for flight information should be displayed. | | |
| **Dependencies:** None | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Ruixi He 27/03/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** This system should provide a Staff with a GUI to switch seat for two exist customers in one flight. | | |
| **Rationale:** A staff wants to switch the seat for two exist customers in a flight. | | |
| **Source:** Staff | | |
| **Fit Criterion:** The booking record of the customer should be updated. | | |
| **Dependencies:** None | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Ruixi He 27/03/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** Any customer should be able to search for a flight | | |
| **Rationale:** A customer must be able to have the liberty of finding the best flight for them | | |
| **Source:** Non-member, Member, Agent | | |
| **Fit Criterion:** The user successfully selects a flight | | |
| **Dependencies:** None | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Sandon Joubert 1/4/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** The system should provide a staff with a GUI to book the flight for the customer | | |
| **Rationale:** A staff assists the customer to book the flight ticket | | |
| **Source:** Staff | | |
| **Fit Criterion:** The ticket can be book successful and the record was saved. | | |
| **Dependencies:** None | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Ruixi He 27/03/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** Any customer should be able to book a flight | | |
| **Rationale:** It’s not a very good airline booking system if you can’t book a flight | | |
| **Source:** Non-member, Member, Agent | | |
| **Fit Criterion:** The user successfully books a flight | | |
| **Dependencies:** Login F\_ | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Sandon Joubert 1/4/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** Any customer should be able to change or cancel a booked flight | | |
| **Rationale:** A customer may of made a mistake or change of mind and wishes to change/cancel a flight after booking | | |
| **Source:** Non-member, Member, Agent | | |
| **Fit Criterion:** The user successfully changes / adds in-flight services | | |
| **Dependencies:** Login F\_ | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Sandon Joubert 1/4/2015 | | |

*4.3.1.2 Profile Subsystem*

A Profile Subsystem that manages not only individual passengers and travel agency profiles, but also the profile of the company staffs (Admin, Manager, Staff).

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Requirement type**  Functional | **Use case** |
| **Description**: system should provide a GUI; this GUI is for any user to log in. | | |
| **Rationale:** users need log in his page | | |
| **Source** all users | | |
| **Fit criterion:** users will log in successfully. | | |
| **Dependencies**: none | | |
| **Rank of importance**：Critical | | |
| **Supporting materials**：None | | |
| **History** create by Junyan Fan 27/03/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** A non-member should allow the user to become a new member | | |
| **Rationale:** Helps them keep track | | |
| **Source:** Non-member | | |
| **Fit Criterion:** The user successfully becomes a non-member | | |
| **Dependencies:** None | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Sandon Joubert 1/4/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** Administrators should be provided with a GUI to create user groups. | | |
| **Rationale:** A administrator wants to create user groups. | | |
| **Source:** Administrator | | |
| **Fit Criterion:** User groups can be successfully created. | | |
| **Dependencies:** Login F\_ | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Zheli Jiang 1/4/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** Administrators should be provided with a GUI to promote staff to manager. | | |
| **Rationale:** A administrator wants to promote staff to manager. | | |
| **Source:** Administrator | | |
| **Fit Criterion:** Staff can be successfully promoted to manager. | | |
| **Dependencies:** Login F\_ | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Zheli Jiang 1/4/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** Administrators should be provided with a GUI to disable accounts. | | |
| **Rationale:** A administrator wants to disable accounts. | | |
| **Source:** Administrator | | |
| **Fit Criterion:** Accounts can be successfully disabled. | | |
| **Dependencies:** Login F\_ | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Zheli Jiang 1/4/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** Administrators should be provided with a GUI to enable accounts. | | |
| **Rationale:** A administrator wants to enable accounts. | | |
| **Source:** Administrator | | |
| **Fit Criterion:** Accounts can be successfully enabled. | | |
| **Dependencies:** Login F\_ | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Zheli Jiang 1/4/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** This system should provide a Staff with a GUI to search customer’s record. | | |
| **Rationale:** A staff wants to search for customer’s record | | |
| **Source:** Staff | | |
| **Fit Criterion:** A GUI form for customer’s record should be displayed. | | |
| **Dependencies:** None | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Ruixi He 27/03/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Requirement type**  Functional | **Use case** |
| **Description:** system should provide a GUI, which can have search function, manager can use this search other users, and see their information | | |
| **Rationale:** managers want to find someone and find his information do something | | |
| **Source**: manager | | |
| **Fit criterion**: the searched user will be display. | | |
| **Dependencies**: none | | |
| **Rank of importance**：Desirable | | |
| **Supporting materials**：None | | |
| **History** create by Junyan Fan 27/03/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Optional | **Use Case #:** |
| **Description:** The system should allow the user to change their password and other account information | | |
| **Rationale:** Sometimes users need to change information about themselves to better reflect changes in their life | | |
| **Source:** Member, Agent | | |
| **Fit Criterion:** The user successfully changes account information | | |
| **Dependencies:** Login F\_ | | |
| **Rank of Importance:** Medium | | |
| **Supporting Materials:** None | | |
| **History:** Created by Sandon Joubert 1/4/2015 | | |

*4.3.1.3 Service Subsystem*

A Service Subsystem that manages in-flight services such as food and drinks.

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Requirement type**  Functional | **Use case** |
| **Description:** System provide a GUI, which can let service manager choose some meal for the plane | | |
| **Rationale**: service manager need choose some meal for chosen plane | | |
| **Source:** service manager | | |
| **Fit criterion:** the plane will have meal can be chosen. | | |
| **Dependencies:** none | | |
| **Rank of importance**：Critical | | |
| **Supporting materials**：None | | |
| **History** create by Junyan Fan 27/03/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Requirement type**  Functional | **Use case** |
| **Description**: system provide a GUI, which can let service manager choose whether have headphone provide for chosen plane | | |
| **Rationale**: service manager want to decide this plane whether headphone provide. | | |
| **Source**: service manager | | |
| **Fit criterion:** chosen plane will be decision whether have headphone provide. | | |
| **Dependencies:** none | | |
| **Rank of importance**：Critical | | |
| **Supporting materials**：None | | |
| **History** create by Junyan Fan 27/03/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Requirement type**  Functional | **Use case** |
| **Description**: system provide a GUI, which can let service manager choose whether have blanket provide for chosen plane | | |
| **Rationale**: service manager want to decide this plane whether blanket provide. | | |
| **Source**: service manager | | |
| **Fit criterion**: chosen plane will be decision whether have blanket provide. | | |
| **Dependencies**: none | | |
| **Rank of importance**：Critical | | |
| **Supporting materials**：None | | |
| **History** create by Junyan Fan 18/04/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement #:** F\_ | **Requirement Type:** Functional | **Use Case #:** |
| **Description:** Any customer should be able to change or add in-flight services | | |
| **Rationale:** A customer may of made a mistake or change of mind and wishes to change/add in-flight services after booking | | |
| **Source:** Non-member, Member, Agent | | |
| **Fit Criterion:** The user successfully changes / adds in-flight services | | |
| **Dependencies:** Login F\_ | | |
| **Rank of Importance:** Critical | | |
| **Supporting Materials:** None | | |
| **History:** Created by Sandon Joubert 1/4/2015 | | |

*4.3.1.4 Reporting Subsystem*

A Reporting Subsystem to generate various summary reports such as: Passengers Report for a day including the occupancy rate etc., various summary Cashier Reports like total revenue for the day, or weeks etc., Monthly Booking Activity Summary, Daily Booking Activity Summary, etc.

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Requirement type**  Functional | **Use case** |
| **Description:** system have a button, when press it, will have a GUI to see all report  He can see some types about report. | | |
| **Rationale:** manager wants to know which report he got. These report can display the company’s state. | | |
| **Source**: all manager | | |
| **Fit criterion**: all report will be display | | |
| **Dependencies**: none | | |
| **Rank of importance**：Essential | | |
| **Supporting materials**：None | | |
| **History** create by Junyan Fan 27/03/2015 | | |

*4.3.2 Non-functional Requirements*

|  |  |  |
| --- | --- | --- |
| **Requirement:**  **N\_4.3.2\_01** | **Requirement Type:**  Performance | **Use Case# :** |
| **Description:** The login process should not take longer than 10 seconds. | | |
| **Rationale:** User login quickly. | | |
| **Source:** System Admin, Manager, Member | | |
| **Fit Criterion:** The login process take in 10 seconds. | | |
| **Dependencies:** None | | |
| **Rank of importance:** Essential | | |
| **Supporting Materials:** None | | |
| **History:** Created by Ruixi He, Zheli Jiang 03/25/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement:** | **Requirement Type:**  Performance | **Use Case# :** |
| **Description:** The system must not take longer than 10 seconds to generate each of the analysis. | | |
| **Rationale:** A user wants to receive results quickly. | | |
| **Source:** System Admin, Manager, Member | | |
| **Fit Criterion:** The System generate each of the analysis in 10 seconds. | | |
| **Dependencies:** None | | |
| **Rank of importance:** Essential | | |
| **Supporting Materials:** None | | |
| **History:** Created by Ruixi He, Zheli Jiang 03/25/2015 | | |

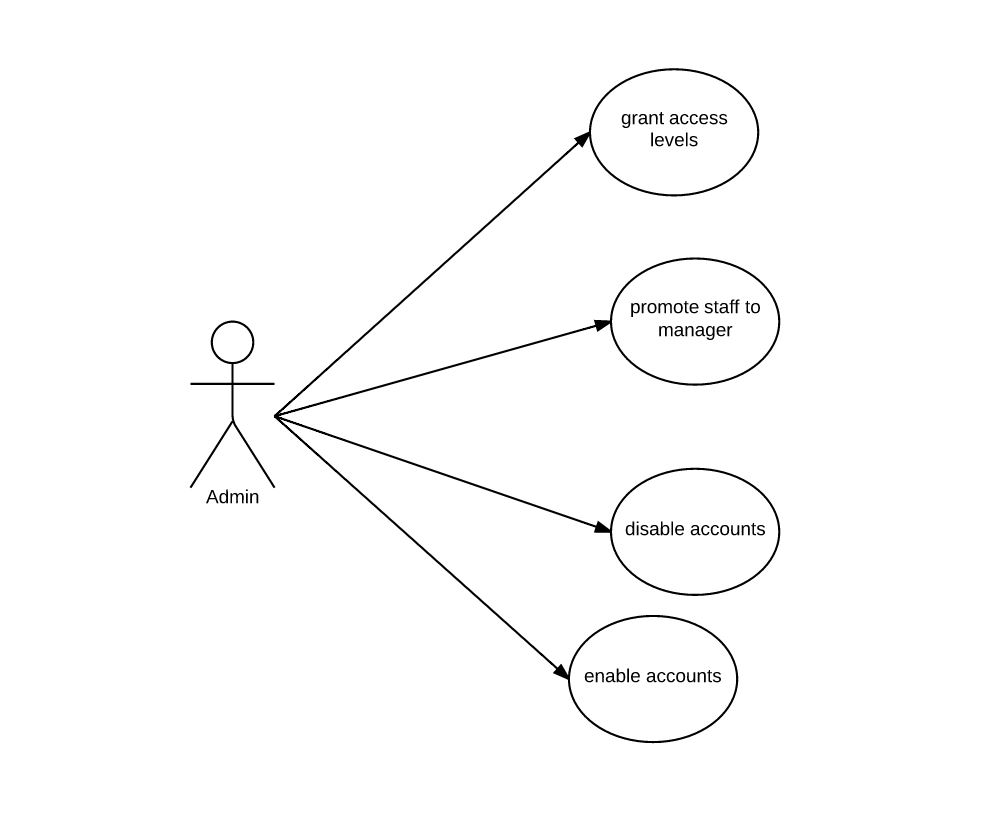
|  |  |  |
| --- | --- | --- |
| **Requirement:** | **Requirement Type:**  Usability | **Use Case# :** |
| **Description:** The system should support GUI in English and other languages. | | |
| **Rationale:** A user wants to use the system in his mother language. | | |
| **Source:** System Admin, Manager, Member | | |
| **Fit Criterion:** The system GUI should be displayed in the language set by every user. | | |
| **Dependencies:** None | | |
| **Rank of importance:** Essential | | |
| **Supporting Materials:** None | | |
| **History:** Created by Ruixi He, Zheli Jiang 03/25/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement:** | **Requirement Type:** | **Use Case# :** |
| **Description:** The system should have user manual to instruct the users to use the system. | | |
| **Rationale:** To help user use the system. | | |
| **Source:** System Admin, Manager, Member | | |
| **Fit Criterion:** The system should have user manual to instruct the users to use the system. | | |
| **Dependencies:** None | | |
| **Rank of importance:** Essential | | |
| **Supporting Materials:** None | | |
| **History:** Created by Ruixi He, Zheli Jiang 03/25/2015 | | |

|  |  |  |
| --- | --- | --- |
| **Requirement:** | **Requirement Type:**  Security | **Use Case# :** |
| **Description:** The system should automatically log out after 5 minutes of being idle and ask the user to re-log in. | | |
| **Rationale:** To control authorized usage of the system. | | |
| **Source:** System Admin, Manager, Member | | |
| **Fit Criterion:** The system should automatically log out after ten minutes of being idle. | | |
| **Dependencies:** None | | |
| **Rank of importance:** Desirable | | |
| **Supporting Materials:** None | | |
| **History:** Created by Ruixi He, Zheli Jiang 03/25/2015 | | |

**5. Use Cases**

5.1 Admin



|  |  |
| --- | --- |
| **Name:** Grant access levels | **ID:AD\_1** |
| **Stakeholders and goals:** Admin - wants to create user groups. | |
| **Description:** Admin needs to create user groups to run the system. | |
| **Actors:** Admin | |
| **Trigger:** Create user groups | |
| **Normal flow:**   * Grant different access levels to users. | |
| **Sub-flows:** None. | |
| **Alternative/Exceptional flows:** None. | |

|  |  |
| --- | --- |
| **Name:** Promote staff to manager | **ID:AD\_2** |
| **Stakeholders and goals:** Admin – wants to promote a staff to manager  Staff – gets promoted to manager | |
| **Description:** Admin can promote staff to manager. | |
| **Actors:** Admin | |
| **Trigger:** Staff is qualified to be promoted. | |
| **Normal flow:**   * Evaluate a staff. * Promote a staff if he/she is qualified. | |
| **Sub-flows:** None. | |
| **Alternative/Exceptional flows:** None. | |

|  |  |
| --- | --- |
| **Name:** Disable accounts | **ID:AD\_3** |
| **Stakeholders and goals:** Admin – wants to disable accounts. | |
| **Description:** Admin can disable accounts | |
| **Actors:** Admin | |
| **Trigger:** Any account behaves suspiciously. | |
| **Normal flow:**   * Disable a suspicious account. | |
| **Sub-flows:** None. | |
| **Alternative/Exceptional flows:** None. | |

|  |  |
| --- | --- |
| **Name:** Enable accounts | **ID:AD\_4** |
| **Stakeholders and goals:** Admin – wants to enable accounts. | |
| **Description:** Admin can enable accounts. | |
| **Actors:** Admin | |
| **Trigger:** Admin enables accounts. | |
| **Normal flow:**   * Enable a disabled account. | |
| **Sub-flows:** None. | |
| **Alternative/Exceptional flows:** None. | |

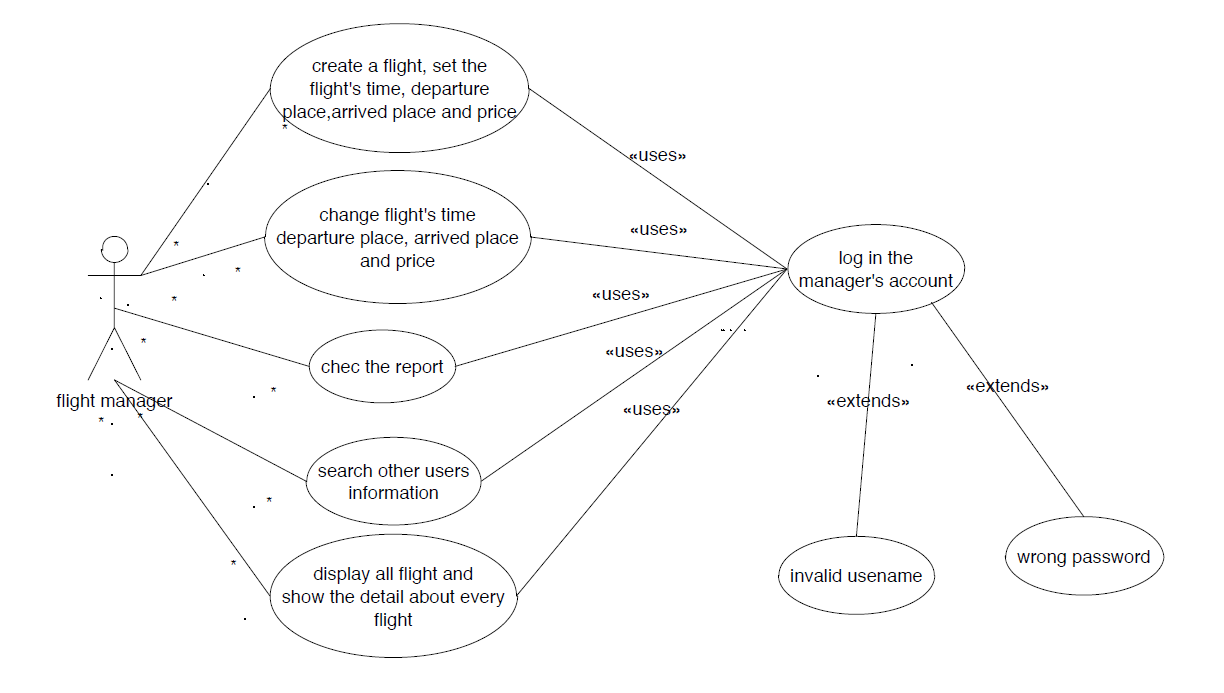
5.2 Manager

|  |  |
| --- | --- |
| **Name:** check report | **ID: M\_1** |
| **Stakeholder and goals:** manager can see report if he have. | |
| **Description:** managers want to see his got report. | |
| **Actor:** managers | |
| **Trigger:** manager wants to check his report. | |
| **Normal flow:**   1. Use his username and password log in his account 2. He presses report button. | |
| **Sub-flow:** none | |
| **Alternative/Exceptional flows:** none. | |

|  |  |
| --- | --- |
| **Name:** search user | **ID:M\_2** |
| **Stakeholder and goals:** manager want to search user | |
| **Description:** flight managers want to get someone’s detail, he will use system to find him | |
| **Actor:** managers | |
| **Trigger:** manager want to get someone’s detail | |
| **Normal flow:**  1. Use his username and password log in his account  2. He presses a find user’s button  3. He put name and id, and click search | |
| **Sub-flow:** none | |
| **Alternative/Exceptional flows:**  1. Invalid name or id. | |

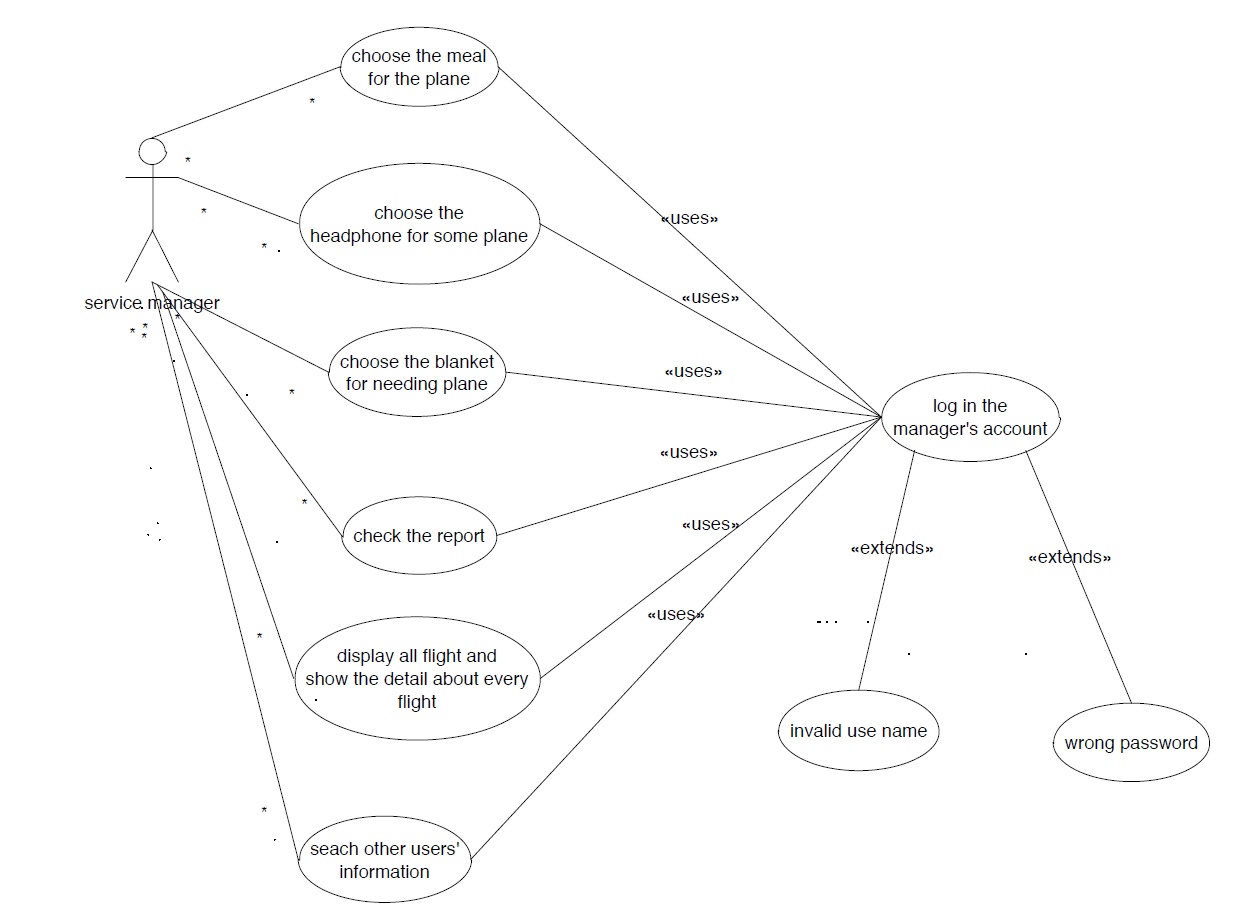
|  |  |
| --- | --- |
| **Name:** display all flight | **ID: M\_3** |
| **Stakeholder and goals:** manager wants to check all flight’s status. | |
| **Description:** managers want to see all fight’s status; he can go in and see its detail. | |
| **Actor:** managers | |
| **Trigger:** manager want to find a flight | |
| **Normal flow:**   1. Use his username and password log in his account 2. He presses display all planes’ button. | |
| **Sub-flow:** none | |
| **Alternative/Exceptional flows:** none | |

*5.2.1 Flight Manager*



|  |  |
| --- | --- |
| **Name:** create a new flight | **ID: FM\_1** |
| **Stakeholder and goals:** flight manager want to create a new flight | |
| **Description:** flight managers want to create a new flight and set the flight’s time, departure place, arrived place and price | |
| **Actor:** flight manager | |
| **Trigger:** flight manager think company need have more flight | |
| **Normal flow:**   1. Use his username and password log in his account 2. He presses a create a new flight button 3. He set all detail about that flight 4. He pressed save button. | |
| **Sub-flow:** none | |
| **Alternative/Exceptional flows:**  1. Some details do not fill. | |

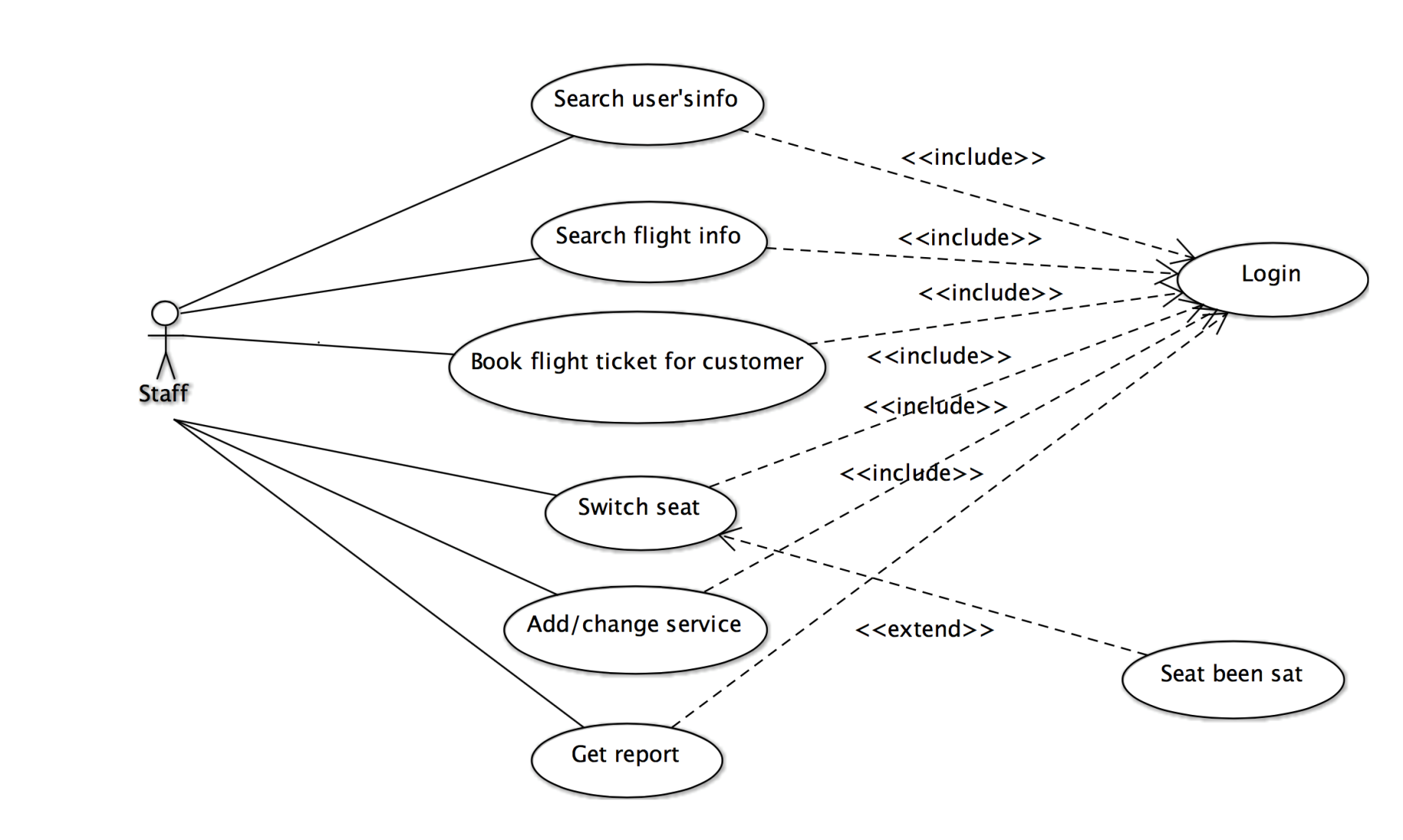
|  |  |
| --- | --- |
| **Name:** modify flight | **ID: FM\_2** |
| **Stakeholder and goals:** flight manager want to modify flight detail | |
| **Description:** flight managers want to modify flight detail and change the flight’s time, departure place, arrived place and price | |
| **Actor:** flight manager | |
| **Trigger:** flight manager think some flights need to modify | |
| **Normal flow:**   1. Use his username and password log in his account 2. He presses a see all flight button 3. Click a flight, and go to this flight detail 4. He change some details about that flight 5. He pressed save button. | |
| **Sub-flow:** none | |
| **Alternative/Exceptional flows:**  1. Some details do not fill. | |

*5.2.2 Service Manager*

|  |  |
| --- | --- |
| **Name:** choose meal | **ID: SM\_1** |
| **Stakeholder and goals:** service manager want to choose the meal for the plane | |
| **Description:** service managers want to choose meal for some planes | |
| **Actor:** service manager | |
| **Trigger:** service manager want to choose some meal for different planes | |
| **Normal flow:**   1. Use his username and password log in his account 2. He presses display plane button 3. He chooses one plane. 4. He chooses some meal for this plane 5. He pressed save button. | |
| **Sub-flow:** none | |
| **Alternative/Exceptional flows:**  1. None | |

|  |  |
| --- | --- |
| **Name:** choose headphone | **ID: SM\_2** |
| **Stakeholder and goals:** service manager want to choose the headphone for the plane | |
| **Description:** service managers want to choose headphone for some planes | |
| **Actor:** service manager | |
| **Trigger:** service manager want to choose headphone for different planes | |
| **Normal flow:**   1. Use his username and password log in his account 2. He presses display plane button 3. He chooses one plane. 4. He chooses headphone for this plane 5. He pressed save button. | |
| **Sub-flow:** none | |
| **Alternative/Exceptional flows:**  1. None | |

|  |  |
| --- | --- |
| **Name:** choose blanket | **ID: SM\_3** |
| **Stakeholder and goals:** service manager want to choose the blanket for the plane | |
| **Description:** service managers want to choose blanket for some planes | |
| **Actor:** service manager | |
| **Trigger:** service manager want to choose blanket for different planes | |
| **Normal flow:**   1. Use his username and password log in his account 2. He presses display plane button 3. He chooses one plane. 4. He chooses blanket for this plane 5. He pressed save button. | |
| **Sub-flow:** none | |
| **Alternative/Exceptional flows:**  1. None | |

5.3 Staff

|  |  |
| --- | --- |
| **Name: Search customer’s record** | **ID: ST\_1** |
| **Stakeholders and goals:** Staff – wants to search customer’s record | |
| **Description:** A staff wants to search customer’s record from the Flight Management System | |
| **Actors:** Staff | |
| **Trigger:** The staff wants to search customer’s record by using the Flight Management System | |
| **Normal flow:**   1. The staff search customer’s record by enter customer’s full name or customer ID 2. The system displays the customer’s record in details | |
| **Sub-flows:**  S1. The staff login | |
| **Alternative/Exceptional flows:**  None | |

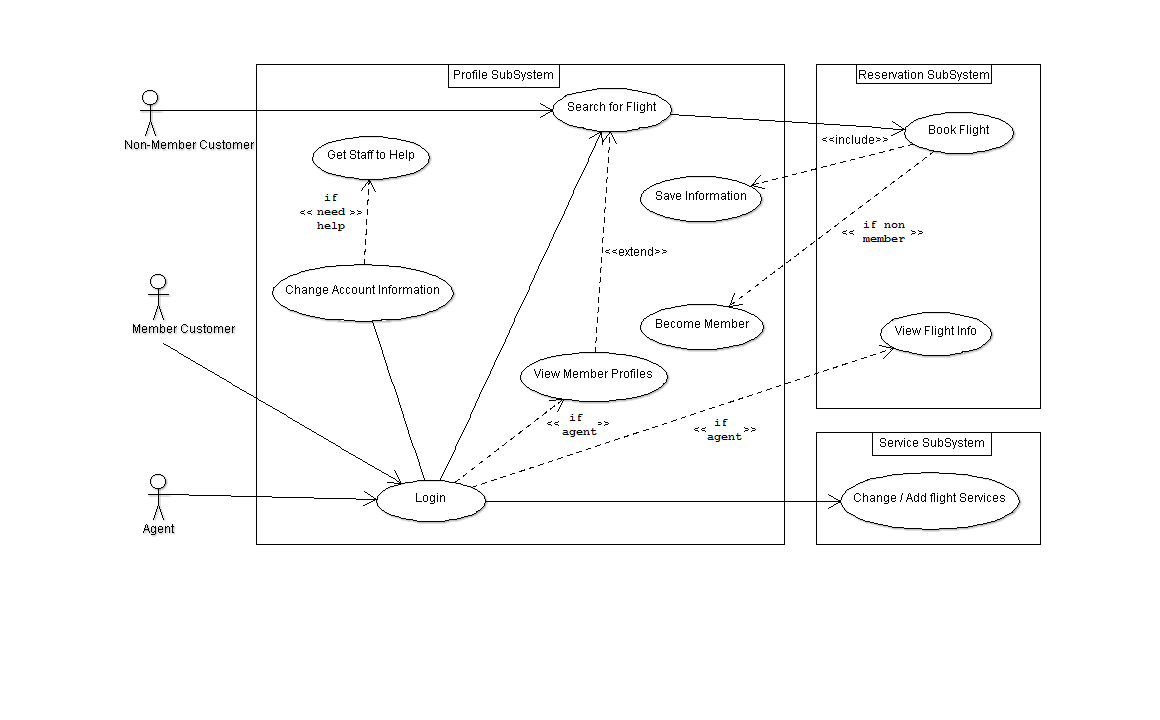
|  |  |
| --- | --- |
| **Name: Search flight info** | **ID: ST\_2** |
| **Stakeholders and goals:** Staff – wants to search flight info | |
| **Description:** A staff wants to search flight info from the Flight Management System | |
| **Actors:** Staff | |
| **Trigger:** The staff wants to search flight info by using the Flight Management System | |
| **Normal flow:**   1. The staff search flight info by enter the origin, destination and date 2. The system displays all the flight info in list 3. The staff choose one flight 4. The system displays the details of that flight | |
| **Sub-flows:**  S1. The staff login to the system | |
| **Alternative/Exceptional flows:**  None | |

|  |  |
| --- | --- |
| **Name: Book flight** | **ID: ST\_3** |
| **Stakeholders and goals:** Staff – wants to search book flight for customers | |
| **Description:** A staff wants to book flight for customers | |
| **Actors:** Staff | |
| **Trigger:** A customer ask the staff to book the flight for he/she | |
| **Normal flow:**  1. | |
| **Sub-flows:**  S1. The staff login | |
| **Alternative/Exceptional flows:**  None | |

|  |  |
| --- | --- |
| **Name: Book flight** | **ID: ST\_4** |
| **Stakeholders and goals:** Staff – wants to search customer’s record | |
| **Description:** A staff wants to search customer’s record from the Flight Management System | |
| **Actors:** Staff | |
| **Trigger:** The staff wants to search customer’s record by using the Flight Management System | |
| **Normal flow:**   1. The staff search customer’s record by enter customer’s full name or customer ID 2. The system displays the customer’s record in details | |
| **Sub-flows:**  S1. The staff login | |
| **Alternative/Exceptional flows:**  None | |

|  |  |
| --- | --- |
| **Name: Book flight** | **ID: ST\_5** |
| **Stakeholders and goals:** Staff – wants to search customer’s record | |
| **Description:** A staff wants to search customer’s record from the Flight Management System | |
| **Actors:** Staff | |
| **Trigger:** The staff wants to search customer’s record by using the Flight Management System | |
| **Normal flow:**   1. The staff search customer’s record by enter customer’s full name or customer ID 2. The system displays the customer’s record in details | |
| **Sub-flows:**  S1. The staff login | |
| **Alternative/Exceptional flows:**  None | |

|  |  |
| --- | --- |
| **Name: Book flight** | **ID: ST\_6** |
| **Stakeholders and goals:** Staff – wants to search customer’s record | |
| **Description:** A staff wants to search customer’s record from the Flight Management System | |
| **Actors:** Staff | |
| **Trigger:** The staff wants to search customer’s record by using the Flight Management System | |
| **Normal flow:**   1. The staff search customer’s record by enter customer’s full name or customer ID 2. The system displays the customer’s record in details | |
| **Sub-flows:**  S1. The staff login | |
| **Alternative/Exceptional flows:**  None | |

5.4 General Public

|  |  |
| --- | --- |
| **Name:** Login | **ID: GP\_1** |
| **Stakeholders and goals:** Member, Agent, Staff, Admin – would like to use the system so must login | |
| **Description:** The user enters in a username and password to login to the system | |
| **Actors:** Member, Agent, Staff, Manager, Admin | |
| **Trigger:** Start up the program | |
| **Normal Flow:**  1. The user enters E-mail address  2. The user enters Password  3. The user clicks login  4. The system checks username / password combination  5. The system successfully logs in the user | |
| **Sub-flows:**  None | |
| **Alternative/Exceptional Flows:**  2a. The user clicks ‘Forgot Password’  2b. The system sends e-mail to user containing their password  4a. The system realises a bad username / password combination  4b. The system notifies user with on screen display that their information is incorrect | |

|  |  |
| --- | --- |
| **Name:** Become Member | **ID: GP\_2** |
| **Stakeholders and goals:** Non-member – register to the system so they may use it | |
| **Description:** User enters details to create an account on the system | |
| **Actors:** Non-member | |
| **Trigger:** Click on register new account | |
| **Normal Flow:**  1. The user enters e-mail  2. The user enters password  3. The user registers account  4. The system sends user with account details and notifies user of success | |
| **Sub-flows:**  None | |
| **Alternative/Exceptional Flows:**  1a. The system notifies the user that the username is taken  1b. The user enters a different username  2a. The system notifies the user that the e-mail is already in use  2b. The user enters a different E-mail | |

|  |  |
| --- | --- |
| **Name:** Change account information | **ID: GP\_3** |
| **Stakeholders and goals:** Member, Agent, Staff, Manager, Admin – Would like to change their account information | |
| **Description:** User can alter either password, e-mail or other details but not their username | |
| **Actors:** Member, Agent, Staff, Manager, Admin | |
| **Trigger:** Click on change account information | |
| **Normal Flow:**  1. The user changes any account information they may wish to change (password and passenger details used when booking flights)  2. The user clicks to save changes  3. The system checks changes and notifies users on success | |
| **Sub-flows:**  pre-1. Login | |
| **Alternative/Exceptional Flows:**  3a. The user may enter invalid information and is prompted accordingly  3b. The user enters valid information | |

|  |  |
| --- | --- |
| **Name:** Search for Flight | **ID: GP\_4** |
| **Stakeholders and goals:** Customers – Would like to look for the flight they wish to use | |
| **Description:** Search for a flight by time, departure city and destination city | |
| **Actors:** Non-member, Member, Agent | |
| **Trigger:** Click search flight | |
| **Normal Flow:**  1. The user click to search for flights  2. The system displays a form of which to pick a date, destination and departure cities.  3. The user enters day they would like to depart on  4. The user enters city of departure  5. The user enters destination city  6. The user clicks search  7. The system displays all matching flights for specified time, destination and departure cities as well as alternative times within certain timeframe of specified time | |
| **Sub-flows:**  None | |
| **Alternative/Exceptional Flows:**  None | |

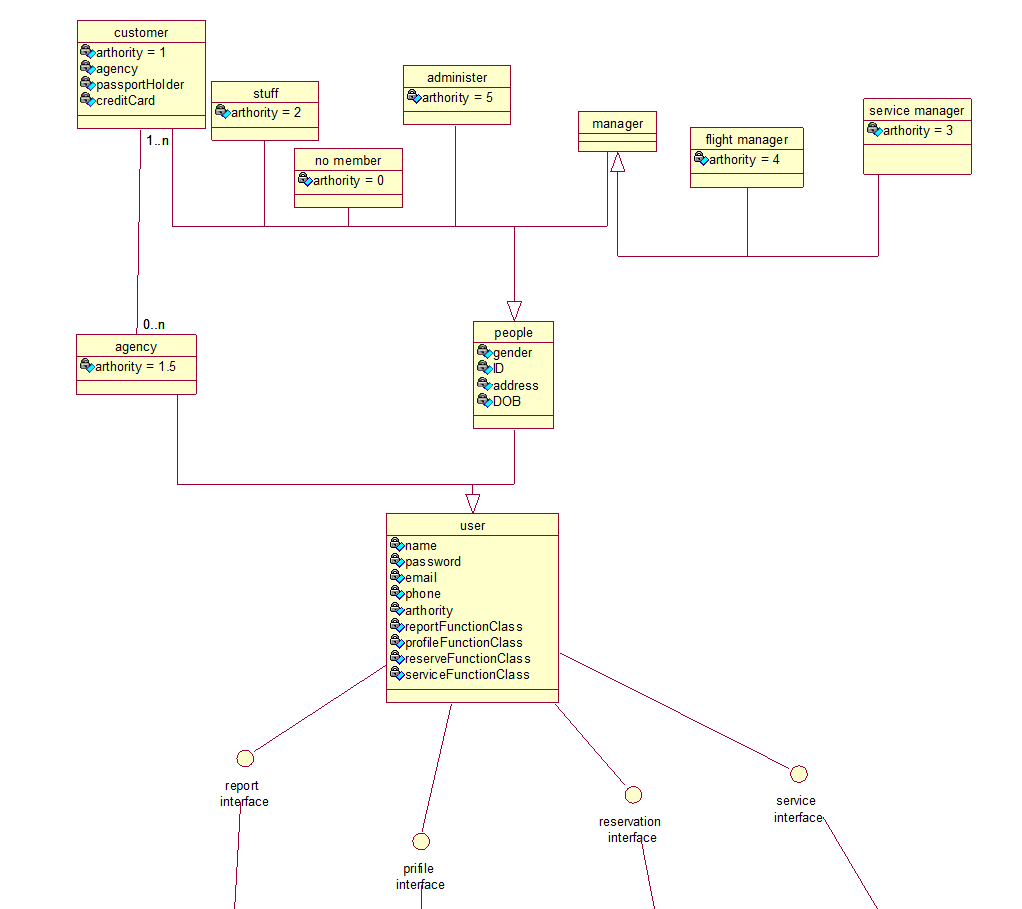
|  |  |
| --- | --- |
| **Name:** Book Flight | **ID: GP\_5** |
| **Stakeholders and goals:** Customers – to place a booking for a flight | |
| **Description:** Customer searches for flight and then selects a flight to book | |
| **Actors:** Member, Agent | |
| **Trigger:** Selects flight from displayed flights received in search flight | |
| **Normal Flow:**  1. The user selects flight they wish to book  2. The system brings up list of options (seat selection, in-flight services, passenger details (which include users account details as first passenger))  3. The user may select a seat they want  4. The user may select any in-flight services they want  5. The user fills in any additional passengers information  6. The user selects if they want any extra baggage  7. The system notifies the user of total price including any services purchased and displays a confirm booking button  8. The user confirms booking  9. The system sends a confirmation e-mail to user and notifies user of successful booking | |
| **Sub-flows:**  Pre-1. Login  Pre-1. Search Flight | |
| **Alternative/Exceptional Flows:**  None | |

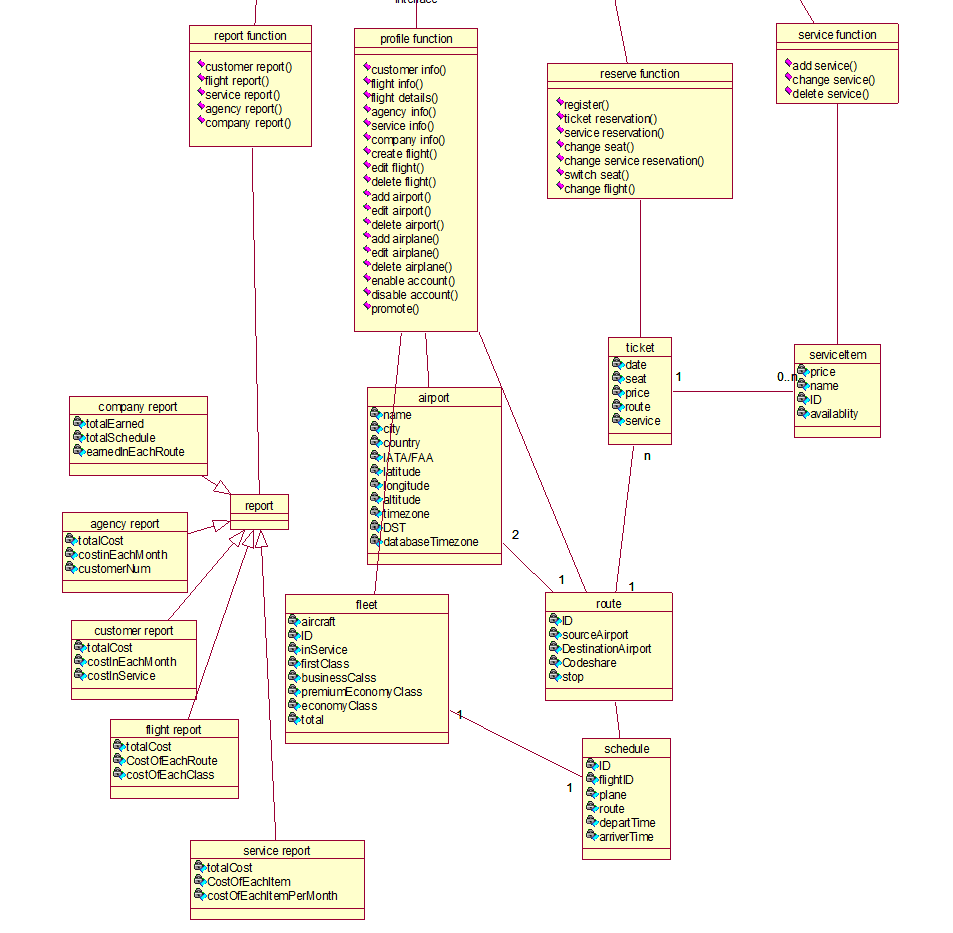
|  |  |
| --- | --- |
| **Name:** Change / Add In-Flight Services | **ID: GP\_6** |
| **Stakeholders and goals:** Customers – to change in-flight services associated with a booking | |
| **Description:** Customer wishes to add or change services to their previously booked flight | |
| **Actors:** Member, Agent | |
| **Trigger:** Selects the change in-flight services button | |
| **Normal Flow:**  1. The user changes the in-flight services  1a. The user adds a new service  1b. The user changes details about an already purchased service  2. The system saves changes and charges or refunds required amount to user | |
| **Sub-flows:**  Pre-1. Login | |
| **Alternative/Exceptional Flows:**  None | |

|  |  |
| --- | --- |
| **Name:** Cancel Flight | **ID: GP­\_7** |
| **Stakeholders and goals:** Customers – to cancel a flight already been booked | |
| **Description:** Customer – cancels a flight booking | |
| **Actors:** Member, Agent | |
| **Trigger:** Selects cancel booking button | |
| **Normal Flow:**  1. The customer selects a booking they wish to cancel  2. The system asks for confirmation that they wish to cancel booking  3. The customer confirms they wish to cancel booking  4. The system refunds customer if booking cancelled 48 hours prior to flight | |
| **Sub-flows:**  Pre-1. Login | |
| **Alternative/Exceptional Flows:**  None | |

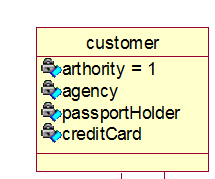
**6. Domain Model**

6.1 Class Diagram





6.2 Detailed Description

**Class Name:** Customer

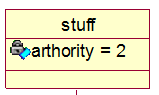
**Super-classes:** People

**Attributes:**

* **Authority:** different level’s authority from 1 to 5, and the customer has level 1 authority.
* **Agency:** Agency name, empty is no agency.
* **PassportHolder:** True or False.
* **CreditCard:** 16-digit unique number.

**Methods:**

None

**Class Name:** Staff

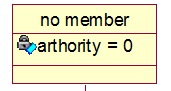
**Super-classes:** People

**Attributes:**

* **Authority:** different level’s authority from 1 to 5, and the staff has level 2 authority.

**Methods:**

None

**Class Name:** NonMember

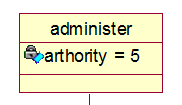
**Super-classes:** People

**Attributes:**

* **Authority:** different level’s authority from 1 to 5, and Non-Member has level 0 authority.

**Methods:**

None

**Class Name:** Administrator

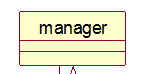
**Super-classes:** People

**Attributes:**

* **Authority:** different level’s authority from 1 to 5, and Admin has level 5 authority.

**Methods:**

**Class Name:** Manager

**Super-classes:** People

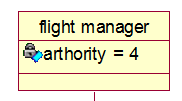
**Sub-classes:** Flight Manager, Service Manager

**Attributes:**

None

**Methods:**

None

**Class Name:** Flight Manager

**Super-classes:** Manager

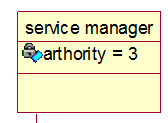
**Attributes:**

* **Authority:** different level’s authority from 1 to 5, and the Flight Manager has level 4 authority.

**Methods:**

None

**Class Name:** Service Manager

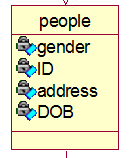
**Super-classes:** Manager

**Attributes:**

* **Authority:** different level’s authority from 1 to 5, and the Service Manager has level 3 authority.

**Methods:**

None

**Class Name:** People

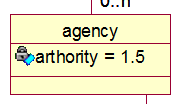
**Super-classes:** User

**Attributes:**

* **Gender:** Male, Female or Other
* **ID:** 8-digit unique number
* **Address:** Home address
* **DOB:** Date of birth, DD-MM-YYYY

**Methods:**

None

**Class Name:** Agency

**Super-classes:** User

**Attributes:**

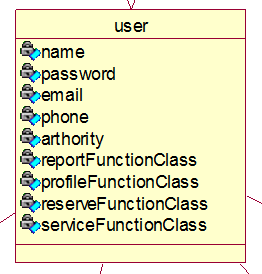
* **Authority:** different level’s authority from 1 to 5, and the Agency has level 1.5 authority.

**Methods:**

None

**Class Name:** User

**Super-classes:** None

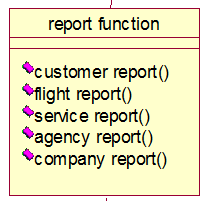
**Attributes:**

* **Name:** user name
* **Password:** Over 8-digit character
* **Email:** Email address
* **Phone:** 10-digit unique number
* **Authority:** Access level, float number
* **ReportFunctionClass:** an interface implement all function in sub-system
* **ProfileFunctionClass:** an interface implement all function in sub-system
* **ReserveFunctionClass:** an interface implement all function in sub-system
* **ServiceFunctionClass:** an interface implement all function in sub-system

**Methods:**

None

**Class Name:** ReportFunction

**Super-classes:** None

**Attributes:**

None

**Methods:**

1. **Customer\_report()**

Allows us to get the customer report

1. **Flight\_report()**

Allows us to get the flight report

1. **Service\_report()**

Allows us to get the service report

1. **Agency\_report()**

Allows us to get the agency report

1. **Company\_report()**

Allows us to get the company report

**Class Name:** Report

**Super-classes:** None

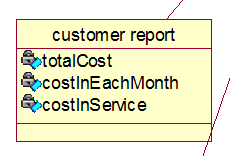
**Sub-classes:** customer\_report, flight\_report(), service\_report(), agency\_report(), company\_report()

**Attributes:**

None

**Methods:**

None

**Class Name:** customer\_report

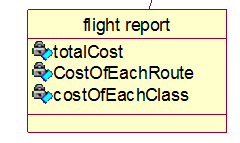
**Super-classes:** Report

**Attributes:**

1. **totalCost:** The total amount spent from customers.
2. **costInEachMonth:** The total amount spent from customers per month.
3. **costInService:** The amount spent on services.

**Methods:**

None

**Class Name:** flight\_report

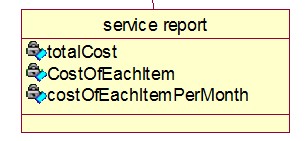
**Super-classes:** Report

**Attributes:**

1. **totalCost:** The total amount spent to a flight.
2. **costOfEachRoute:** The total amount spent to a flight of each route.
3. **costOfEachClass:** The total amount spent to each fare class.

**Methods:**

None

**Class Name:** service\_report

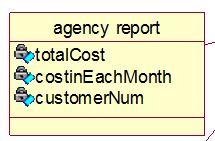
**Super-classes:** Report

**Attributes:**

1. **totalCast:** The total amount spent to service.
2. **costOfEachItem:** The total amount spent to each service item.
3. **costOfEachItemPerMonth:** The total amount spent to each service item per month.

**Methods:**

None

**Class Name**: agency\_report

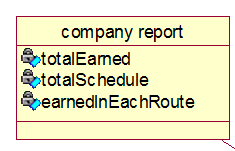
**Super-classes**: Report

**Attributes:**

1. **totalCost:** The total amount spent from an agency.
2. **costInEachMonth:** The total amount spent an agency each month.
3. **customerNum:** The total amount of customs an agency have.

**Methods:**

None

**Class Name:** company\_report

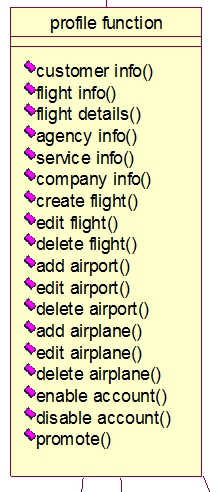
**Super-classes:** Report

**Attributes:**

1. **totalEarned:** The total amount the company earned.
2. **totalSchedule:** The total amount of schedule company have.
3. **earnedInEachRoute:** The total amount earned from each route.

**Methods:**

None

**Class Name:** Profile\_Function

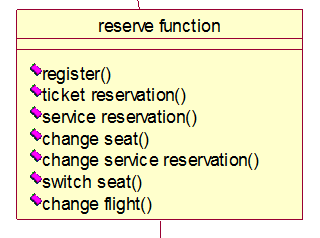
**Super-classes:** None

**Attributes:** None

**Methods:**

1. **customer\_info():** Get customer info, such as name, credit card, reservation…
2. **flight\_info():** Get flight information, such as destination, source airport...
3. **flight\_details():** Get more information about flight, such as seat be reserved, left ticket in each class…
4. **agency\_info():** Get agency information, such as agency name, member in this agency, ticket reserved by this agency…
5. **service\_info():** Get service information, such as service company provided…
6. **company\_info():** Get company information, such as airplane number, airport can be arrived…
7. **create\_flight():** Create a new airline.
8. **edit\_flight():** Change the information of a flight.
9. **delete\_flight():** Delete an exist flight.
10. **add\_airport():** Add a new airport could be arrived.
11. **edit\_airport():** Change the information of an airport.
12. **delete\_airport():** Delete an exist airport.
13. **add\_airplane():** Add a new aircraft.
14. **edit\_airplane():** Change the information of an aircraft.
15. **delete\_airplane():** Delete an exist aircraft.
16. **enable\_account():** Enable account
17. **disable\_account():** Disable account
18. **promote():** Promote a stuff to manager

**Class Name:** Reserve\_Function

**Super-classes:** None

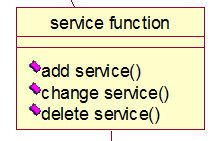
**Attributes:**

None

**Methods:**

1. **register():** Gather Non-member’s information; create a member account which could make reservation.
2. **ticket\_reservation():** Book a ticket
3. **service\_reservation():** Add new service to a ticket
4. **change\_seat():** Select of reselect a seat if available.
5. **change\_service\_reservation():** Delete or change exist service.
6. **switch\_seat():** Stuff can switch passages’ seat after them select a seat.
7. **change\_flight():** Rescheduling a ticket

**Class Name:** Service\_Function

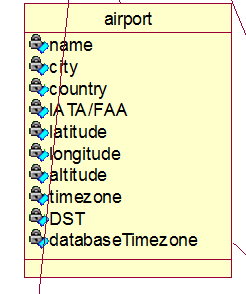
**Super-classes:** None

**Attributes:**

None

**Methods:**

1. **add\_service():** Create a new service, make it selectable
2. **change\_service():** Change exist service’s name, price, availability…
3. **delete\_service():** Delete a exist service.

**Class Name:** Airport

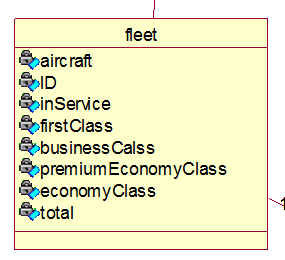
**Super-classes:** none

**Attributes:**

* **name:** airport name
* **city:** city name
* **country:** country name
* **IATA/FAA:** organization belonged
* **Latitude:** float number
* **Longitude:** float number
* **Altitude:** float number
* **Timezone:** timezone code
* **DST:** Daylight Saving Time
* **databaseTimezone:** timezone code

**Methods:**

None

**Class Name:** Fleet

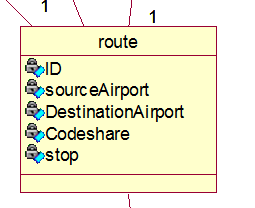
**Super-classes:** none

**Attributes:**

* **aircraft:** name of the model
* **ID:** 8-digit unique number
* **inService:** Yes or No, is this in fleet
* **firstClass:** seat number in first class
* **businessClass:** seat number in bussiness class
* **premiumEconomyClass:** seat number in premium economy class
* **economyClass:** seat number in economy class
* **total:** total seat number

**Methods:**

None

**Class Name:** Route

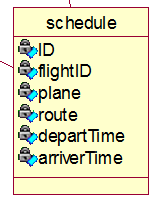
**Super-classes:** none

**Attributes:**

* I**D:** 8-digit unique number
* **sourceAirport:** airport ID
* **destinationAirport:** airport ID
* **codeshare:** Yes or No, is share route code with other company
* **stop:** integer number

**Methods:**

None

**Class Name:** Schedule

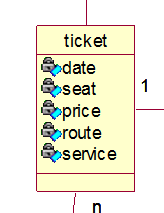
**Super-classes:** none

**Attributes:**

* **ID:** 8-digit unique number
* **flightID:** flight ID
* **plane:** fleet ID
* **route:** route ID
* **departTime:** hh:mm DD-MM-YYYY
* **arrivalTime:** hh:mm DD-MM-YYYY

**Methods:**

None

**Class Name:** Ticket

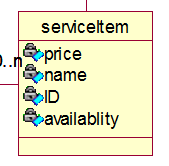
**Super-classes:** none

**Attributes:**

* **date:** DD-MM-YYYY
* **seat:** seat number
* **price:** ticket price and service prive
* **route:** route number
* **service:** service item list

**Methods:**

None

**Class Name:** ServiceItem

**Super-classes:** none

**Attributes:**

* **price:** float number
* **name:** service name
* **id:** unique number
* **availability:** international, all, domestic or none

**Methods:**

None

**7. Meta-report**

7.1 Tabular Summary of the Group Structure

|  |  |  |
| --- | --- | --- |
| Group Members | Roles | Artifacts Delivered |
| Ruixi He | Project Manager  Business-Process Analyst  Requirement Specifier  System Analyst | * Distribute work * Make project plan * Finish business case * Create GitHub repository * Provide functional requirements of Staff * Provide use cases of Staff and detail description * Record meeting agendas * Finish individual work diary |
| Siyuan Hou | Lead Programmer  Software Engineer | * Class diagram Architectural design * Individual work diary |
| Zheli Jiang | System Analyst  Business Designer  Requirement Specifier | 1. Detailed plan 2. Gantt Chart 3. Meeting agendas 4. Admin-related use cases 5. Admin-related use case descriptions 6. Admin-related functional requirements 7. Individual work diary |
| Junyan Fan | System Analyst  Business Designer  Requirement Specifier | * Manager use cases * Manager use case descriptions * Manager functional requirements * Individual work diary |
| Sandon Joubert | System Analyst  Business Designer  Requirement Specifier | * Risk and counter measures * General public use cases * General public use case descriptions * General public functional requirements * Individual work diary |

7.2 Group Meeting Summary

*7.2.1 Report of Group Meeting1*

Date: 13/03/2015

Time: 2:30pm - 4:30pm

Meeting called by Attendance: Ruixi He - Project Manager,

Zheli Jiang,

Junyan Fan,

Siyuan Hou

Place: #204 Library

Preparation:

-None

Meeting Agendas:

-Introduction: The group members introduced themselves to each other and shared thoughts about on this project.

-We spent 15 minutes reading specification and we discussed about what the system should look like (implementation) and what should our report includes (documentation).

-Implementation: We are to develop a flight management system that has several subsystems that we need to implement respectively. This Flight Management System includes four major subsystems:

* 1. Reservation Subsystem that can add, change and modify all flight reservations, seat selection, ticketing, flight availability, flight details, rates and conditions.
  2. Profile Subsystem that manages individual passengers and travel agency profiles.
  3. Service Subsystem that manages in-flight services.
  4. Reporting Subsystem to generate various summary reports.

-Documentation: The documentation includes two major parts. This is the first time we encounter this sort of documentation so we are unfamiliar with the structure of the report. Hence we decided to have some research on SRS.

-We drew a user group architecture to show the client.

-In order to fully understand the objectives of this assignment, we clarified some questions as listed below:

* 1. Can we build the system based on website and java?
  2. What kind of users to use this system?
  3. What's this system for, a company or the platform?
  4. Reservation phase: Does customer have to login to do reservation?
  5. What's different between normal customer and agency, do we need to think about how the agency uses this system?
  6. Service: What's the service system for, for customer or the management?
  7. Reporting: What kind of data should we provide different users? (Different data?)

-Risks:

-Initially the group had only four members. Furthermore, none of the members speak English as first language. So documentation appeared challenging at first.

-One of our members had no experience coding Java, which could be a potential risk.

-Roles distribution:

-Ruixi He was assigned project manager because he did this subject twice so he was very experienced. Also he is very talented in leading.

-Siyuan Hou was assigned lead programmer because he had most experience in coding (JAVA) and he had a good understanding of this system structure.

-Zheli Jiang was assigned documentation manager.

-Junyan Fan was assigned design manager.

Work to do:

-Do some research on SRS.

-Consult the clients.

*7.2.2 Report of Group Meeting2*

Date: 20/03/2015

Time: 2:30pm - 4:30pm

Meeting called by Attendance: Ruixi He - Project Manager,

Zheli Jiang,

Junyan Fan,

Siyuan Hou

Place: #204 Library

Preparation:

-Research on SRS.

-Answers to questions:

1.Can we build the system base on website and java?

Yes.

2.What kind of users to use this system?

User who use the system are: General (Public), Administrator (super user they can do everything), Staff (they help users do things), Managers (a flight manager and aa service manager, they set prices and stuff)

3.What's this system for, a company or the platform?

Just one airline.

4.Reservation phase: Does customer have to login to do reservation?

Customer does not have to log in can have separate member and customer.

5.What's different between normal customer and agency, do we need to think about how the agency use this system?

Difference between agent and customer, travel agents get discount for customer.

6.Service: What's the service system for, for customer or the management?

When a customer books flight they can choose their food and drink in booking process.

7.Reporting: What kind of data should we provide different users? (different data?)

Various reports would like to see travel agent reports like how many people they get through them.

Meeting Agendas:

-A new member joined our team who speaks English as first language.

-We decided to use RUP as our development model.

-Sort out the stakeholders to elicit requirements.

-We improved our user group architecture after consultation with the client.

-General public

-Customer

-Normal Customer

-Member

-Agency

-Administrator

-Staff

-Manager

-Flight Manager

-Service Manager

Work to do:

-Zheli Jiang: Admin requirements

-Ruixi He: Staff requirements

-Junyan Fan: Manager requirements

-Sandon Joubert: General public requirements

-Siyuan Hou: System Analysis

*7.2.3 Report of Group Meeting3*

Date: 27/03/2015

Time: 2:30pm - 4:30pm

Meeting called by Attendance: Ruixi He - Project Manager,

Zheli Jiang,

Junyan Fan,

Siyuan Hou,

Sandon Joubert

Place: #204 Library

Preparation:

-Zheli Jiang: Admin requirements

-Ruixi He: Staff requirements

-Junyan Fan: Manager requirements

-Sandon Joubert: General public requirements

-Siyuan Hou: System Analysis

Meeting Agendas:

-This is the first time Sandon joined our meeting so we updated him for our group progress.

-Gather and discuss the requirements.

1.Admin:  
1. Grant different access levels(admin, staff, manager, general public)  
2. Promote staff to manager  
3. Enable/Disable accounts  
4. View all profiles and flight info  
5. Change flight details(departure time,etc.)

2.Manager:  
Flight Manager:  
1. Create a flight(dep/arr, time, plane type, price, flight number, etc.)  
2. Change flight details(departure time,etc.).  
Service Manager:  
1. Order meals

3.Staff:  
1. Switch seats  
2. Search for users involved in this flight and filter the results  
Search for flight info(Staff can see more specific info than customers)

4.General Public:  
Customer:  
Non-member:  
1. Enter details during first purchase and automatically become registered  
Member:  
1. Discount  
Agency:  
1. Further discount (than members)  
2. View profiles of users who bought tickets through this agency  
3. Search for flight info(pretty much the same as staff)  
4. Save user profiles(list)

Report:(Reports of numbered items will be generated)  
Admin:  
1. Financing;  
2. Staff;  
3. Customer;  
4. Sales volume.

Manager:  
Flight Manager:  
1. Flight (those under his/her management)  
Service Manager:  
1. Meal

Staff:  
1. Flight and meal

General Public:  
Customer:  
1. Purchase history  
Agency:  
1. User profiles who bought tickets through this agency

-Siyuan Hou provided a sketch of class diagram.

Work to do:

-Develop the requirements.

-Improve class diagram.

-Create use cases.

*7.2.4 Report of Group Meeting4*

Date: 03/04/2015

Time: 2:30pm - 4:30pm

Meeting called by Attendance: Ruixi He - Project Manager,

Zheli Jiang,

Junyan Fan,

Siyuan Hou,

Sandon Joubert

Place: #204 Library

Preparation:

-All use cases and class diagram.

-All improved functional requirements,

Meeting Agendas:

-Discuss the use cases.

-We created scenarios which include both successful and unsuccessful situations according to our individual actors and use cases.

-Ruixi He had another assignment due so he failed to complete his use cases. Therefore he had to finish his use cases during the meeting.

-We decided to use GitHub as our version control software. (Before we only used Facebook to contact and share files.) We all registered and shared our files on GitHub.

-Siyuan Hou proposed an idea that we can have an interface with access level passed in as an argument to identify user groups and which functions they have access to.

Work to do:

-Ruixi He: Business Case

-Zheli Jiang: Detailed Plan

-Junyan Fan: Use case development

-Siyuan Hou: Class diagram improvement

-Sandon Joubert: Risk and counter measure

*7.2.5 Report of Group Meeting5*

Date: 17/04/2015

Time: 2:30pm - 4:30pm

Meeting called by Attendance: Ruixi He - Project Manager,

Zheli Jiang,

Junyan Fan,

Siyuan Hou,

Sandon Joubert

Place: #204 Library

Preparation:

-Ruixi He: Business Case

-Zheli Jiang: Detailed Plan

-Junyan Fan: Use case development

-Siyuan Hou: Class diagram improvement

-Sandon Joubert: Risk and counter measure

Meeting Agendas:

-We combined the use cases and requirements.

-We have a discussion on meta-report.

Work to do:

-Finish SRS.

7.3 Individual Work Diaries

*7.3.1 Ruixi He:*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week | Start Date | Date Planed to Finish | Tasks | Description | Completed Date |
| 2 | 13/03/2015 | 13/03/2015 | Read project specification | We spent 15 minutes to go through the project specification. | 13/03/2015 |
| 13/03/2015 | 13/03/2015 | Discussion about the project | We discussed and wrote down what kind of system should we build, and what should the report have. Then we got that we need to build a flight management system which include four major subsystem: Reservation Subsystem (manages all fight reservations), Profile Subsystem (manages users profile), Service Subsystem (manages in-flight service) and Reporting Subsystem (generate summary reports). | 13/03/2015 |
| 13/03/2015 | 13/03/2015 | Prepare questions to ask client | After we need to prepare some questions to ask the client to elicit the requirements of the system. | 13/03/2015 |
| 13/03/2015 | 13/03/2015 | Define risks | We defined the potential risks not only about the creation of the system, but also our team. | 13/03/2015 |
| 13/03/2015 | 13/03/2015 | Assign roles to each group member | I was at the role as a leader at this time because I have more experience, Siyuan Hou was assigned lead programmer because he had most experience in coding (JAVA) and he had a good understanding of this system structure,  Zheli Jiang was assigned documentation manager,and  Junyan Fan was assigned design manager. | 13/03/2015 |
| 13/03/2015 | 20/03/2015 | Research about SRS | From the meeting till to next meeting, we need to do some research and understand what is SRS and how to make a good SRS | 15/03/2015 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week | Start Date | Date Planed to Finish | Tasks | Description | Completed Date |
| 3 | 20/03/2015 | 20/03/2015 | Discussion about the team structure | Because we got a new group member, Sandon, from this week lab, so we need to reassign the work again. And Sandon had thing to do, so he did not attend this week’s meeting. | 20/03/2015 |
| 20/03/2015 | 20/03/2015 | SRS | We talk about what we got from the research, and create the format of the SRS. | 20/03/2015 |
| 20/03/2015 | 20/03/2015 | Discuss about the answers we got from last client meeting | From the answers we got from last client meeting, we decide to build this system based on website and Java as background. We also correct the misunderstands that we had when we go through the specification, this is very helpful to the analysis of the requirements. | 20/03/2015 |
| 20/03/2015 | 20/03/2015 | Define stakeholders | We defined the stakeholders and potential of the system, and we got four major users group: Admin, Manager, Staff and General Public. | 20/03/2015 |
| 20/03/2015 | 27/03/2015 | Assign work to do before next meeting | We need to create the first draft of the requirements:  Zheli Jiang: Admin requirements,  Ruixi He: Staff requirements,  Junyan Fan: Manager requirements,  Sandon Joubert: General public requirements,  Siyuan Hou: System Analysis | 27/03/2015 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week | Start Date | Date Planed to Finish | Tasks | Description | Completed Date |
| 4 | 27/03/2015 | 27/03/2015 | Review before meeting | This is the first time Sandon joined in the group meeting, so we need to tell him the whole progress of project and what we had done till now. | 27/03/2015 |
| 27/03/2015 | 27/03/2015 | Analyses the requirements | Each group member presented their part of the requirements, and we need to find whether there are faults or missing. | 27/03/2015 |
| 27/03/2015 | 27/03/2015 | System structure | Our lead programmer showed us the system structure by the present the Domain Model. | 27/03/2015 |
| 27/03/2015 | 03/04/2015 | Assign work to do before next meeting | We need to continue the development of requirements, and also Siyuan Hou still needs to do more works to improve the class diagram.  And the rest members need to make the draft of the use cases. And also prepare the questions to ask the client on next lab. | 03/04/2015 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week | Start Date | Date Planed to Finish | Tasks | Description | Completed Date |
| 5 | 03/04/2015 | 03/04/2015 | Review before meeting | From the meeting of the client on last lab, we showed them the draft of use cases that we got, and the client pointed out the question we had, like the non-member cannot book the flight until he/she register an account in the system, and the use cases of service manager should be more specific. | 03/04/2015 |
| 03/04/2015 | 03/04/2015 | Make scenarios of the use cases | Each group member present two scenarios which include both successful and unsuccessful situations. | 03/04/2015 |
| 03/04/2015 | 03/04/2015 | Version control system (GitHub) | We prefer to use GitHub as our version control system, I created a repository (CSCI222) and ask each member to share their file on the repository. | 03/04/2015 |
| 03/04/2015 | 03/04/2015 | System Structure | Siyuan Hou proposed an idea that we can have an interface with access level passed in as an argument to identify user groups and which functions they have access to. | 03/04/2015 |
| 03/04/2015 | 03/04/2015 | Assign work to do before next meeting | Next week should be the Easter break, which means we need to do more work during the break, so each members should keep developing the functional requirements and also finish the report, and here is the plan:  -Ruixi He: Business Case  -Zheli Jiang: Detailed Plan  -Junyan Fan: Use case development  -Siyuan Hou: Class diagram improvement    -Sandon Joubert: Risk and counter measure | 17/04/2015 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week | Start Date | Date Planed to Finish | Tasks | Description | Completed Date |
| 6 | 17/04/2015 | 17/04/2015 | Review before meeting | Each member showed their work together. | 17/04/2015 |
| 17/04/2015 | 17/04/2015 | Combine works together | Combine all the requirements and use cases together, and find whether there are any requirements or use cases missing. | 17/04/2015 |
| 17/04/2015 | 17/04/2015 | Meta-report | Create the roles of the group, and combine each meeting record to form the summary of the meeting report, each member can finish their work diaries base on the meeting report. | 17/04/2015 |
| 17/04/2015 | 17/04/2015 | First draft of project report | Group each thing to form the first draft of report, and we need to do more work to complete the final report before the due day. | 21/04/2015 |

*7.3.2 Zheli Jiang:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| week | Start Date | Date Planned to Finish | Tasks | Complete Date |
| 2 | 13/03/2015 | 14/03/2015 |  Sort out basic functional requirements. | 14/03/2015 |
| 14/03/2015 | 16/03/2015 |  Draw up a user group structure. | 16/03/2015 |
| 16/03/2015 | 18/03/2015 |  Clarify some questions related to specification. | 18/03/2015 |
| 18/03/2015 | 19/03/2015 |  Record meeting agendas. | 19/03/2015 |
| 19/03/2015 | 20/03/2015 |  Research on SRS. | 20/03/2015 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| week | Start Date | Date Planned to Finish | Tasks | Complete Date |
| 3 | 20/03/2015 | 22/03/2015 |  Improve the user group structure. | 22/03/2015 |
| 22/03/2015 | 25/03/2015 |  Work out administrator requirements | 25/03/2015 |
| 25/03/2015 | 25/03/2015 |  Consult with client. | 25/03/2015 |
| 25/03/2015 | 27/03/2015 |  Work out who our stakeholders could be. | 27/03/2015 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| week | Start Date | Date Planned to Finish | Tasks | Complete Date |
| 4 | 27/03/2015 | 27/03/2015 |  Discussed requirements with members.   Gather the requirements. | 27/03/2015 |
| 27/03/2015 | 31/03/2015 |  Improve class diagram. | 31/03/2015 |
| 31/03/2015 | 03/04/2015 |  Create administrator use cases. | 03/04/2015 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| week | Start Date | Date Planned to Finish | Tasks | Complete Date |
| 5 | 10/04/2015 | 10/04/2015 |  Do some research on GitHub and decide to use it as version control software. | 10/04/2015 |
| 10/04/2015 | 11/04/2015 |  Upload files to GitHub. | 11/04/2015 |
| 11/04/2015 | 17/04/2015 |  Complete business case. | 17/04/2015 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| week | Start Date | Date Planned to Finish | Tasks | Complete Date |
| 6 | 17/04/2015 | 18/04/2015 |  Finish meta-report. | 18/04/2015 |
| 18/04/2015 | 19/04/2015 |  Finish individual work-diary. | 19/04/2015 |
| 19/04/2015 | 20/04/2015 |  Tidy Up SRS. | 20/04/2015 |

*7.3.3 Junyan Fan:*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week | Start date | Date plan to finish | Task | Complete date | |
| 2 | 13/03/2015 | 13/03/2015 | Read assignment specification. | 13/03/2015 | |
| 13/03/2015 | 13/03/2015 | Discuss this system’s structure. | 13/03/2015 | |
| 13/03/2015 | 13/03/2015 | Prepare question for lab | 15/03/2015 | |
| In this week, we have meeting in Friday, so we finish lots of tasks in meeting, because this system is easy to understand, so I have lots of ideas for this design. | | | | |
| 3 | 20/03/2015 | 20/03/2015 | Everyone team member have their part for this assignment | 20/03/2015 | |
| 20/03/2015 | 20/03/2015 | Discuss about how to start the assignment | 20/03/2015 | |
| 20/03/2015 | 20/03/2015 | Discuss who is stakeholder | 20/03/2015 | |
| 20/03/2015 | 24/03/2015 | Write use case about manager | 25/03/2015 | |
| In this week, we got part for everyone, so we have work to do, my part is manager, I use some time to think about it. I finish it before next lab, because I can get advice from teacher. | | | | | |
| 4 | 27/03/2015 | 27/03/2015 | Discuss every member’s use case | 27/03/2015 | |
| 27/03/2015 | 27/03/2015 | Give some advice for class diagram | 27/03/2015 | |
| 27/03/2015 | 27/03/2015 | Discuss requirement for every actor | 27/03/2015 | |
| 27/03/2015 | 30/03/2015 | Write requirement about manager | 1/04/2015 | |
| This week we discuss our use case and discuss about the requirement, after meeting, I start to write requirement. | | | | | |
| 5 | 03/04/2015 | 03/04/2015 | Discuss everyone’s requirement | 03/04/2015 | |
| 03/04/2015 | 03/06/2015 | Modify my use case | 06/04/2015 | |
| 03/04/2015 | 03/07/2015 | Modify my requirement | 07/04/2015 | |
| In this week lab, I get lots of advice from teacher about use case, so I modify my use case and requirement. I get a final vision. | | | | | |
| 6 | 17/4/2015 | 20/4/2015 | Write use case description | 21/4/2015 | |
| 17/4/2015 | 21/4/2015 | Write my daily | 21/4/2015 | |
| In this week, I use my use case and write use case description. After those work, I finish my daily and ask manager whether have other work, and do some other work to help manager finish this assignment | | | | | |

*7.3.4 Siyuan Hou:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| week | Start Date | Date Planned to Finish | Tasks | Complete Date |
| 2 | 13/03/2015 | 14/03/2015 | discuss basic system purpose | 14/03/2015 |
| 14/03/2015 | 16/03/2015 | decide role of group member | 16/03/2015 |
| 16/03/2015 | 18/03/2015 | discuss unknown problem | 18/03/2015 |
| 18/03/2015 | 19/03/2015 | record meeting content. | 19/03/2015 |
| 19/03/2015 | 20/03/2015 | did some research about java and C++ | 20/03/2015 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| week | Start Date | Date Planned to Finish | Tasks | Complete Date |
| 3 | 20/03/2015 | 22/03/2015 | decide system language | 22/03/2015 |
| 22/03/2015 | 25/03/2015 | discuss administrator requirements | 25/03/2015 |
| 25/03/2015 | 25/03/2015 | clear special requirements with client. | 25/03/2015 |
| 25/03/2015 | 27/03/2015 | research features about java | 27/03/2015 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| week | Start Date | Date Planned to Finish | Tasks | Complete Date |
| 4 | 27/03/2015 | 27/03/2015 | implement system structure | 27/03/2015 |
| 27/03/2015 | 31/03/2015 | implement class diagram. | 31/03/2015 |
| 31/03/2015 | 03/04/2015 | discuss sub-system, and get new idea about system implement | 03/04/2015 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| week | Start Date | Date Planned to Finish | Tasks | Complete Date |
| 5 | 10/04/2015 | 10/04/2015 | create GitHub | 10/04/2015 |
| 10/04/2015 | 11/04/2015 | improve system structure | 11/04/2015 |
| 11/04/2015 | 17/04/2015 | improve class diagram | 17/04/2015 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| week | Start Date | Date Planned to Finish | Tasks | Complete Date |
| 6 | 17/04/2015 | 18/04/2015 | research about JSP | 18/04/2015 |
| 18/04/2015 | 19/04/2015 | discuss report issues | 19/04/2015 |
| 19/04/2015 | 20/04/2015 | complete individual work-diary. | 20/04/2015 |

*7.3.5 Sandon Joubert:*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week | Start Time | Planned End Time | Task | Description | Actual End Time |
| 3 | 20/03/2015 | 27/03/2015 | Analyse General Public Requirements | Did draft use case for the General Public, as well as corresponding use case description and srs | 24/03/2015 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week | Start Time | Planned End Time | Task | Description | Actual End Time |
| 4 | I was away for a weeding for most of this week |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week | Start Time | Planned End Time | Task | Description | Actual End Time |
| 5 | 03/04/2015 | 03/04/2015 | Complete Requirements | Finalised used case diagrams, description and srs | 03/04/2015 |
| 03/04/2015 | 03/04/2015 | Analyse Risk Management | Analysed and finalised possible risks and countermeasures | 06/04/2015 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week | Start Time | Planned End Time | Task | Description | Actual End Time |
| 6 | 10/04/2015 | 17/04/2015 | Individual Work Diary | Bring my individual work diary up to standard and organise it to be near ready for hand in | 12/04/2015 |
| 10/04/2015 | 23/04/2015 | Class Description | Complete the class description | 22/04/2015 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week | Start Time | Planned End Time | Task | Description | Actual End Time |
| 7 | 17/04/2015 | 23/04/2015 | Documentation | Help out with any final documentation to be handed in at the end of the week | 23/04/2015 |

7.4 Screenshot of Version Control Software (GitHub)

8. Member Contribution Assessment

|  |  |  |
| --- | --- | --- |
| Name/Student number | Contribution Rank | Signature |
| Zheli Jiang/4391457 | Contributed |  |
| Junyan Fan/ | Contributed |  |
| Siyuan Hou/ | Contributed |  |
| Sandon Joubert/ | Contributed |  |
| Ruixi He/4174458 | Contributed |  |

**Appendix**

**Definitions**

**actor**

Someone or something outside the system or business that interacts with the system or business.

**class**

A description of a set of objects that have the same responsibilities, relationships, operations, attributes, and semantics.

**domain**

An area of knowledge or activity characterized by a family of related systems.

**requirement**

A description of a condition or capability of a system; either derived directly from user needs or stated in a contract, standard, specification, or other formally imposed document.

**use case**

A sequence of actions a system performs that yields an observable result of value to a particular actor. A use-case class contains all main, alternate, and exception flows of events related to producing the observable result of value. Technically, a use case is a class whose instances are scenarios.

**vision**

The user's or customer's view of the product to be developed.

**Acronyms**

SRS - Software Requirement Specification

Admin – Project Management System Administrator

GUI - Graphical User Interface

MB - Megabytes