CSc 4350: Software Engineering

Fall 2016

Team Phoenix

Fergus Kelley

Andrew Truong

Ryan Ocampo

Deividas Rutkauskas

Viraj Shah

Document #5 — Project Rationale

October 13th, 2016

Rationales

***Project Topic***

When we formed Team Phoenix for this semester project, we want to pick a topic that would challenge our critical thinking and problem solving skills, while at the same time resembling a project we might encounter in our careers. We decided to create a piece of resort management software because it would require careful thinking to fully understand the needs of hotel guests and employees. From there, we decided to include additional tools for employee to manage other aspects that might be necessary at a resort, such as conference rooms and a restaurant. We want to explore the complexity of the business operations and create a piece of software that is both powerful and easy to use.

We are targeting the software to be used by the hotel employees to manage the operations of the resort. The software can be utilized on any contemporary operating system, such as Windows, Mac OS, or Linux since it will be developed using the Java language. A local database will be used for all persistent data necessary for the software.

***Use Cases***

We developed use cases to clearly specify how users might interact with the software. These use cases and rationales formed the basis for the front-end of the software:

1, User Login: To allow only authorized employees to access the system.

2, Create Profile: To manage employee profiles to ensure correct access to employees with proper credentials as well as keeping employee information updated.

3, Manage Hotel Rooms: To provide flexibility in the number and type of hotel rooms available. The software is designed to accommodate resorts with differing numbers of hotel rooms.

4, Modify Room Profile: To provide flexibility in the types of rooms available. Rooms may differ in their amenities offered.

5, Create Hotel Reservation: To allow employees to find available rooms fulfilling certain criteria and create reservations for any given room and timespan.

6, Modify or Output Room Billing: To allow employees to accurately bill reservations and additional services

7, Manage Room Maintenance: To allow employees to document and update the maintenance status of any hotel room.

8, Create Event Room Profiles: To provide flexibility in the number and types of event room available. The software is designed to accommodate resorts with differing numbers and types of event rooms.

9, Event Booking: It is necessary for event rooms to be easily booked.

10, Event Billing: To allow event rooms must be able to be billed effectively.

11, Unified Ordering System: To enabled a variety of orders and services for serving guests of the resort. This includes room service, restaurant service, and others.

12, Modify or Update Order: To allow orders to be tracked and updated as necessary.

13, Output Orders: To output a list of in-progress orders, filtered by certain criteria, so that employees can complete and update them.

14, Manage Restaurant: To allow changing and updating of the restaurant menu according to customer preferences and business needs.

15, Generate Reports: To provide updated real time information about operations of the result to ensure efficiency and profitability.

***Function Point Cost Analysis***

Software engineering requires understanding the costs of development. By using function point cost analysis, we can roughly estimate the cost of developing any given piece of software based on its requirements and complexity. This allows developers to make smarter decisions about future development and provide appropriate time and costs estimates to clients.

***Object Design***

Classes were developed for the following components and objects: employees, hotel rooms, hotel room reservations, inventory, used inventory, invoice, billed items, event rooms, event room booking, event room reservations, restaurant items, in-resort restaurant, and catered meal items. The reason we distinguished the hotel room class and the reservation is because one class considers the attributes of the room, and the other considers the billing and amount of time the room can be reserved. For the sake of organization and ease, we concurred that this organization satisfies those principles. Employee objects are created so as to login with their profiles to access the functionalities of the system. We distinguished used inventory from initial inventory for the sake of documenting reports and statistics behind replenishing the hotel’s resources. The invoice will contain all the costs that the guest has accumulated in a data structure, so in that way we can merely create a billable item object to add to said invoice object. We decided that guests can also book event rooms to hold for special occasions at extra costs. Looking up the information of the room and the process of booking the room are once again separate like the hotel room and reservation object so to keep the distinct information separate to avoid confusion. Initially we thought that an ordering system that can take different kinds of requests would be convenient, but we considered the different kinds of orders that we can get. Thus, we divided orders into maintenance orders, cleaning orders, and food service orders mainly. Finally, classes were considered for keeping track of data concerning an in-resort restaurant. On the assumption that the restaurant offers catering in addition, the menu items offered in restaurant and the catered items that sends out would be handled differently, given they have different attributes. The restaurant itself will be an object that contains data that keeps track of the processes that go on inside the restaurant. The restaurant information management would be a challenging addition to the other hotel on-goings we consider in creating the hotel management system.

Requirements Traceability Matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entry # | Para # | PRMS Requirements Traceability Matrix | Type | Use Case |
| 1 | 2.0 | The PRMS shall provide a graphic user interface (GUI) with tabs. | SW |  |
| 2 | 2.0 | Each tab of the interface shall profile functionality for employees to interact with the PRMS. | SW |  |
| 3 | 2.0 | When the employee clicks on a tab, the PRMS shall display a new interface with the appropriate functionality for that tab. | SW |  |
| 4 | 3.0 | The PRMS shall provide a tab of the interface for creating and managing employee profiles | SW |  |
| 5 | 3.0 | The employee profiles shall be stored in a local database. | SW |  |
| 6 | 3.0 | Each profile shall contain information about the employee. | SW |  |
| 7 | 3.0 | Each profile will contain information about the employee’s access to the PRMS (authentication). | NTH |  |
| 8 | 3.1 | The PRMS shall provide an interface where the new employee profiles are created. | SW |  |
| 9 | 3.1 | Each employee profile shall contain the employee’s name, job title, unique ID number, username, and password. | SW |  |
| 10 | 3.1 | The PRMS shall allow the entry of the employee’s profile information into the new profile window. | SW | UC2 |
| 11 | 3.1 | The PRMS shall create the employee’s unique ID number with cannot be changed. | SW |  |
| 12 | 3.1 | The PRMS will provide functionality for suitably authorized employees to modify the information contained in already existing profiles. | SW | UC2 |
| 13 | 3.2 | The PRMS will limit access by way of a username and password | SW | UC1 |
| 14 | 3.2 | The username and password will be created when the employee profile is created. | SW |  |
| 15 | 3.2 | The employee profile will contain information about which interface tabs are available to each employee based on the employee’s title. | SW |  |
| 16 | 3.2 | The PRMS will log employee interactions with the PRMS in a text file. | NTH |  |
| 17 | 4.0 | The PRMS shall allow the management and reservation of the resort’s hotel rooms. | SW | UC3 |
| 18 | 4.1 | The PRMS shall provide a tab of the user interface for managing the resort’s hotel rooms. | SW | UC3 |
| 19 | 4.1 | The PRMS shall provide functionality for describing the number of floors of the hotel, and the number of rooms on each floor. | SW | UC3 |
| 20 | 4.1 | The PRMS shall assign each room a number based on which floor of the hotel the room is located. | SW |  |
| 21 | 4.1 | The PRMS shall provide functionality to enter and modify default attributes about each room. | SW | UC4 |
| 22 | 4.1 | Each room profile shall have attributes and a list of the rooms inventory. | SW |  |
| 23 | 4.1 | For each item in the room, the room inventory shall keep track of the name, quantity, expected quantity, and whether the item is consumable. | SW |  |
| 24 | 4.1 | The PRMS shall keep track of the number of each item in the rooms inventory that has been replaces. | SW |  |
| 25 | 4.1 | The room profiles shall be stored in a local database. | SWC |  |
| 26 | 4.2 | The PRMS shall provide a tab of the user interface that provides functionality for employees to create and modify reservations for each room. | SW | UC5 |
| 27 | 4.2 | The room reservations shall be stored in a local database. | SWC |  |
| 28 | 4.2 | Each reservation shall contain attributes. | SW |  |
| 29 | 4.2 | The reservation’s billing information shall contain fields. | SW |  |
| 30 | 4.2 | The PRMS shall allow employees to query the list of rooms by the room’s attributes and whether a reservation exists for a given timespan. | SW | UC5 |
| 31 | 4.2 | The PRMS shall allow employees to create reservations for any room that does not have a reservation for a given time span. | SW | UC5 |
| 32 | 4.2 | The PRMS shall require the employee to enter valid information for all the fields in the room reservation information. | SW |  |
| 33 | 4.2 | The PRMS shall allow employees to modify a reservation’s billing information by adding or removing additional charges. | SW | UC6 |
| 34 | 4.2 | The PRMS shall allow employees to modify a reservations timespan to extend or shorten a guest’s stay. | SW | UC5 |
| 35 | 4.2 | The PRMS shall allow employees to output an itemized invoice containing all information about the room, including the calculated cost of the reservation based on the price per night and timespan. | SW | UC6 |
| 36 | 4.3 | The PRMS shall provide a tab of the user interface that provides functionality for employees to manage room maintenance and inventory. | SW | UC7 |
| 37 | 4.3 | The PRMS shall allow the employee to note modify the number of items in the room inventory to indicate whether replacements are needed. | SW | UC7 |
| 38 | 4.3 | The PRMS shall provide an interface to note when the room has been cleaned or the inventory has been replaced and update the necessary fields in the room profile. | SW | UC7 |
| 39 | 4.3 | The PRMS shall allow the employee to output a report of all rooms which haven’t been cleaned in a given time period, rooms which have maintenance notes, or rooms with insufficient inventory. | SW | UC7 |
| 40 | 5.0 | The PRMS shall allow the management and reservation of the resort’s events rooms. | SW |  |
| 41 | 5.1 | The PRMS shall provide a tab of the user interface for managing the resort’s events rooms. | SW | UC8 |
| 42 | 5.1 | The PRMS shall provide functionality for describing the number of events rooms in the resort. | SW | UC8 |
| 43 | 5.1 | The PRMS shall provide functionality to enter and modify default attributes about each events room. | SW |  |
| 44 | 5.1 | Each events room shall have attributes. | SW |  |
| 45 | 5.1 | The events room profiles shall be stored in a local database. | SWC |  |
| 46 | 5.2 | The PRMS shall provide a tab of the user interface that provides functionality for employees to create and modify bookings for events rooms. | SW | UC9, UC10 |
| 47 | 5.2 | Each events room reservation shall contain attributes. | SW |  |
| 48 | 5.2 | The booking billing information shall contain fields. | SW |  |
| 49 | 5.2 | The PRMS shall allow employees to query the list of events rooms by the room’s attributes and whether a booking exists for a given timespan. | SW | UC9 |
| 50 | 6.0 | The PRMS shall provide functionality for employees to manage the various services that are necessary to provide to the guests. | SW |  |
| 51 | 6.1 | The PRMS shall provide a tab of the user interface that provides functionality for employees to create and modify service orders. | SW | UC11, UC12 |
| 52 | 6.1 | Each order shall contain the attributes. | SW |  |
| 53 | 6.1 | The PRMS shall allow the employee to create a new order. | SW | UC11 |
| 54 | 6.1 | For a new order, the state of the order shall default to “ordered” and the date and time expected shall default to the current time. | SW |  |
| 55 | 6.1 | The PRMS shall ensure that the employee has selected or created an invoice for the order provided. | SW |  |
| 56 | 6.1 | The PRMS shall display an interface for entering in the various data for the type of order the employee has selected | SW | UC11 |
| 57 | 6.1 | The PRMS shall allow employees to view a list of orders sorted by time expected and filterable by current state | SW | UC13 |
| 58 | 6.1 | The PRMS shall allow the employee to update the state of the order. | SW | UC12 |
| 59 | 6.1 | Once the state of the order is changed to “delivered,” the invoice associated with the order shall be updated with the name and cost of the order. | SW |  |
| 60 | 6.1 | Orders shall be stored in a local database. | SWC |  |
| 61 | 6.2 | The PRMS shall provide an interface that provides functionality for managing the hotel restaurant. | SW |  |
| 62 | 6.2 | The PRMS shall allow employees to create and modify a list of restaurant items available for room service and table service. | SW | UC14 |
| 63 | 6.2 | Each restaurant item shall contain attributes. | SW |  |
| 64 | 6.2 | The PRMS shall allow employees to create and modify a list of tables available for seating at the restaurant | SW | UC14 |
| 65 | 6.2 | Each table shall contain attributes. | SW |  |
| 66 | 6.2 | When creating an order for room service or table service, the PRMS shall require that the employee select one or more items and select or create an invoice for billing. | SW | UC11 |
| 67 | 6.2 | If creating an order for table service, the PRMS shall require that the employee select a table number from the list of restaurant tables. | SW | UC11 |
| 68 | 6.3 | The PRMS shall allow employees to create and modify a list of options available for catered meals. | SW | UC14 |
| 69 | 6.3 | Each catered option shall contain attributes. | SW |  |
| 70 | 6.3 | When creating an order for catered meals, the PRMS shall require that the employee select a conference room booking for delivery and billing, as well as enter the number of servings required. | SW | UC11 |
| 71 | 6.4 | The PRMS shall provide an interface that provides functionality for creating general orders. | SW | UC11 |
| 72 | 6.4 | When creating a general service order, the PRMS shall provide a text box for entering specific information about the order. | SW | UC11 |
| 73 | 7.0 | The PRMS shall provide a tab of the user interface that provides functionality for employees to easily query the various databases used by the software. | SW | UC15 |
| 74 | 7.0 | The PRMS shall provide interfaces for returning information in any given timespan. | SW | UC15 |

Work Schedule Diagram

Gantt Chart

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Start Date:  August 22, 2016** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Week:** | **1** |  |  |  |  | **2** |  |  |  |  | **3** |  |  |  |  | **4** |  |  |  |  | **5** |  |  |  |  |
|  | **M** | **T** | **W** | **T** | **F** | **M** | **T** | **W** | **T** | **F** | **M** | **T** | **W** | **T** | **F** | **M** | **T** | **W** | **T** | **F** | **M** | **T** | **W** | **T** | **F** |
| **Document 1** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Fergus |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Andrew |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Ryan |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Devidas |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Viraj |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Document 2** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Fergus |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Andrew |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Ryan |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Devidas |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Viraj |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Document 3** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Fergus: Gantt chart, FP, RTM |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Andrew: Use cases and diagrams |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Ryan: Use cases and diagrams |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Devidas: Use cases and diagrams |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Viraj: Use cases and diagrams |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Continued from previous page.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Week:** | **6** |  |  |  |  | **7** |  |  |  |  | **8** |  |  |  |  | **9** |  |  |  |  | **10** |  |  |  |  |
|  | **M** | **T** | **W** | **T** | **F** | **M** | **T** | **W** | **T** | **F** | **M** | **T** | **W** | **T** | **F** | **M** | **T** | **W** | **T** | **F** | **M** | **T** | **W** | **T** | **F** |
| **Document 3** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Fergus: Gantt chart, FP, RTM |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Andrew: Use cases and diagrams |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Ryan: Use cases and diagrams |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Devidas: Use cases and diagrams |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Viraj: Use cases and diagrams |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Document 4** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Fergus: Gantt chart, WSD, RTM |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Andrew: Object design |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Ryan: Object design, Rationale |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Devidas: Object design |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Viraj: Object design |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Document 5** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Fergus: Rationale, Gantt, WSD |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Andrew |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Ryan: Document created |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Devidas |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Viraj |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **Document 6** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Fergus |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Andrew: Source Code |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Ryan: Source Code |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Devidas: Source Code |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Viraj: Source Code |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Dictionary

*Embedded database*: a database that is used within a single application. This is opposed to a client-server database which involve two different software applications communicating.

*Graphical User Interface* (abbreviated *GUI*): A program that allows the user to interact with the computer using icons and other visual indicators.

*Local Database*: a collection of digital indexed information that can be searched, referenced, changed, compared or otherwise manipulated. A local database is stored on the same computer that is used to access it.

Change Log

**Date and Time:** October 11, 2016 — 10:26 PM

**Team member:** Ryan Ocampo

**Description:** Initial creation of document

**Version:** 1 — Current version accumulating changes

**Date and Time:** October 12, 2016 — 7:55 PM

**Team member:** Fergus Kelley

**Description:** Project rationales gathered and updated.

Updated Gantt chart and work schedule diagram.

**Version:** 1 — Current version accumulating changes