**Final Deliverable**

CIS 4911 – Senior Project

Virtual Queue

**Member:**

Kely Cid

**Instructor**

Masoud Sadjadi

**Mentor**

[Bernard Parenteau](http://spws-dev.cis.fiu.edu/senior-project-website-v4/user/231)

**Date**

October 20th 2014

Copyright © Florida International University 2014. All Rights Reserved. No part of the Virtual Queue Project or documentation may be reproduced or modified without the express consent of Florida International University, School of Computing & Information Sciences. Redistribution or commercial use is strictly prohibited.

**Executive Summary**

The Virtual Queue System will be designed for theme parks and other businesses that have multiple rides or events for which patrons typically wait in line. The idea is that both the theme park and the patron would benefit by the patrons walking around the park (and maybe spending money) rather than standing in line.

This document gives an introduction to the Virtual Queue System. Chapter 1 gives a basic introduction about the Virtual Queue System, including the problem definition, scope of the system, overall development methodology, definitions, acronyms, and abbreviations, and overview of document. Chapter 2 introduces from the feasibility study the description of the current system identifying limitations and constrains, the description of the alternative solutions considered and explanations of why the solution was selected. Chapter 3 includes the project organization, project personnel and hardware and software resources, the identification of tasks, milestones and deliverables and costs of project. Chapter 4 presents the system requirements, including functional and no functional requirements and requirement analysis. Chapter 5 introduces the system design, subsystem decomposition, hardware and software mapping, persistent data management, and security and privacy of the VQ system. Chapter 6 is the detailed design chapter including static model, dynamic model and code specification. Chapter 7 specifies the subsystem test, system tests, and evaluation of tests. Chapter 8 defines the domain terms used in the document and Chapter 9 contains the Appendixes from A to H in the following order: Project Schedule, All use cases with non functional requirements, User Interface Designs, Analysis Models, Design Models, Documented Class Interfaces and constrains, Documented code for test drivers and stubs, and diary of meetings. Finally chapter 10 shows all references that were used.

Contents

1. Introduction 6

1.1. Problem definition 6

1.2. Scope of System 6

1.3. Overall Development Methodology 7

1.4. Definitions, Acronyms, and Abbreviations 7

1.5. Overview of document 8

2. Feasibility Study 9

2.1. Description of Current System (Limitations and Constraints) 9

2.2. Description of Alternatives 9

2.3. Recommendation with explanation of why the solution was selected. 9

3. Project Plan 10

3.1. Project Organization 10

3.1.1. Project Personnel Organization 10

3.1.2. Hardware and Software Resources 10

3.2 Identification of Tasks, Milestones and Deliverables (work breakdown) 11

3.3 Cost of the Project 14

4. Proposed System Requirements 14

4.1 Functional and Non-Functional Requirements 15

4.2 Analysis of System Requirements 17

4.2.1 Use case model 18

4.2.3 Static model 18

4.2.4 Dynamic model 19

5. System Design (i.e., overall system design) 19

5.1 Overview 19

5.2 Subsystem Decomposition 20

5.3 Hardware-Software Mapping 20

5.4 Persistent Data Management 21

5.5 Security/Privacy 22

6. Detailed Design 22

6.1 Overview 22

6.2 Static models 22

6.3 Dynamic Model 22

6.4 Code Specification 22

7. System Validation 22

7.1 Subsystem Tests 23

7.2 System Tests 23

7.3 Evaluation of Tests 30

8. Glossary 30

9. Appendix 31

9.1 Appendix A - Project schedule (Gantt chart or PERT Chart) 32

9.2. Appendix B – All use cases with nonfunctional requirements 32

9.3. Appendix C – User Interface Designs 32

9.4. Appendix D – Analysis models (static and dynamic) 32

9.5. Appendix E – Design models (static and dynamic) 32

9.6. Appendix F – Documented class interfaces (code) and constrains 32

9.7. Appendix G – Documented code for test drivers and stubs 32

9.8. Appendix H – Diary of Meetings and Tasks for the entire semester 32

10. References 38

# 1. Introduction

The introductory chapter gives some background information about the Virtual Queue system. In the following sections, the problem definition, and scope of the system will be described along with the overall development methodology. In addition, definitions, acronyms, and abbreviations of terms used in this deliverable will be provided and explained on this chapter. Finally, the chapter will conclude with a brief explanation of what to expect from the following chapters of the document.

## 1.1. Problem definition

When going to a park, or to any other venue that provides multiple recurring rides or events, customers typically wait in line until is time for them to go into the ride or event. This is definitely time consuming, since one could be doing something else like be walking around, buying souvenirs, or food, or going perhaps to another ride or event. By waiting in line, the venue is making money for that specific ride/event, but is losing potential additional sales by customers being in line rather than walking around the venue visiting other areas of the venue.

The creation of the Virtual Queue system is the proposed solution to the problem explained above. The system will provide customers the benefit of enjoying other amenities offered in the venue (including, but not limited to rides, food) instead of waiting in line. In addition, the system will keep information about ride or event time and capacity and allow the customers to sign in to different events or rides. In this way, customers will not have the need to wait for a ride in line to go to another one. Furthermore, customers will be notified as their time on their rides or events approaches. At the same time, the business will take advantage of this system because of the fact that customers will now have the opportunity to either sign in to another ride, or walk around to buy souvenirs, food, or something else offered at the specific place.

## 1.2. Scope of System

The main purpose of the Virtual Queue system is to add functionalities and new features to the current system, which will greatly benefit both the customers and the park/event place. The current system requires the customers to buy their tickets and wait in line at the venue until it’s their turn for their event or ride. At some theme parks, there is not even the possibility of buying the ticket for the desired ride online; it has to be purchased by the customers at the theme park.

The VQ will be created with features that will greatly benefit the customers as well as the theme park or other venue using the system. Customers will now be able to login into the system, browse all rides available at the venue park, choose their preferred ones, and add them to their list of rides. Venue registered users will also be able to get a notification before their selected rides/events starts and remove themselves from a registered ride. They will also be able to see all the queues he/she registered and logout when they decide to do so. All functionalities will be created to facilitate and improve the quality time spend at the park or event by customers, and hopefully the sales of the theme park will increase as expected.

## 1.3. Overall Development Methodology

## 1.4. Definitions, Acronyms, and Abbreviations

**Definitions**

* **Customer**: A person or organization that buys goods or services from a store or business.
* **Theme park:** An amusement park with a unifying setting or idea.
* **Alternative:** A possible manner by which a given problem may be resolved.
* **Task:** A piece of job that serves as a unit of work.
* **Guest Users**: Anyone who wants to browse through the site and view offered deals before placing an order.
* **Registered Users**: Users that have already created an online account and can place orders and view previous orders they have requested.
* **System:** The system itself.

**Acronyms**

- **VQ**: Virtual Queue

- **FIU:** Florida International University

- **SCIS:** School of Computing & Information Sciences

**Abbreviations**

As of right now, there are no abbreviations for this project.

## 1.5. Overview of document

The following chapter of this document will cover general information about the project. Chapter 2 introduces from the feasibility study the description of the current system identifying limitations and constrains, the description of the alternative solutions considered and explanations of why the solution was selected. Chapter 3 includes the project organization, project personnel and hardware and software resources, the identification of tasks, milestones and deliverables and costs of project. Chapter 4 presents the system requirements, including functional and no functional requirements and requirement analysis. Chapter 5 introduces the system design, subsystem decomposition, hardware and software mapping, persistent data management, and security and privacy of the VQ system. Chapter 6 is the detailed design chapter including static model, dynamic model and code specification. Chapter 7 specifies the subsystem test, system tests, and evaluation of tests. Chapter 8 defines the domain terms used in the document and Chapter 9 contains the Appendix, in which miscellaneous information, such as charts and tables are shown. Finally chapter 10 shows all references that were used.

# 2. Feasibility Study

The feasibility study chapter explores the idea of a virtual queue from a practical point of view. Initially, it considers the fact that there is no system developed in charged of completing the desired tasks. In addition, it describes the alternative solution considered and explanation of why the solution was selected.

## 2.1. Description of Current System (Limitations and Constraints)

The current system requires the customers to buy their tickets and wait in line at the venue until it’s their turn for their event or ride. At some theme parks, there is not even the possibility of buying the ticket for the desired ride online; it has to be purchased by the customers at the theme park. There is no mechanism to allow the user to register and sign in to see the available rides and select the rides they want; or to logout when they decide to do so; or to allow registered users to log in and logout. It lacks of a system that allows the user the ability to be added to their selected rides. It does not have a system with the ability to allow the user to see all the queues he/she registered. There is no mechanism for adding/removing registered users from the queue according to the queuing algorithm, or to allow users to remove themselves from a registered ride. There is no system with the capability of notifying the users before their selected rides/events starts.

## 2.2. Description of Alternatives

The alternative for this project is to build a new system from scratch. This will give the ability and control required over the architecture of the system. Furthermore, all features will be implemented from the beginning. There will be no need of modifying any existing code, which will increase the efficiency of programming.

## 2.3. Recommendation with explanation of why the solution was selected.

The alternative described above is the only option available for this project. Taking into account this is the first version of this system, there is no other platform that was provided to modify or extend. Therefore, starting this project implementation from the beginning is the only and best option to develop the system and implement all new functionalities. So, the conclusion is to start building the project from the beginning. Since there is no previous platforms implemented that can be take into consideration, the only and best solution is to start from scratch. The implementation of the new system will provide functionalities that will make the new system easier and better.

# 3. Project Plan

This chapter introduces the VQ system a project management perspective. First of all, the project organization that is going to be used will be described, with all roles listed. Following, hardware and software required in order to develop the project will be listed. Finally, tasks, milestones, and deliverables will also be listed.

## 3.1. Project Organization

For this project, I will be in charge of all functionality and roles of the system.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Roles | Tasks | Periods required | Key Phases |
| Kely Cid | * Project Manager * Developer * Test Engineer * Document Editor | * Project Status * Schedule/Tasks * Test Results * Implementation Integration * Project Documentation (Deliverables) * Any other additional tasks | 9/1/14  to 12/12/14 | All |

### 3.1.1. Project Personnel Organization

All sections of this project will be assigned to the only member of the group, which will be in charge to develop all the components needed to support the different functionalities.

### 3.1.2. Hardware and Software Resources

**Hardware**

The following are the specification of the computer that will be used for the development of this project:

**Laptop with:**

**- Processing Power:** 1.8 GHz Intel Core i5

**- RAM Memory:** 4 GB 1600 MHz DDR3

**- Available space on hard drive:** 128 GB flash storage

**Other Devices**

- **Input devices**

a) Keyboard- Standard keyboard

b) Mouse- Trackpad mouse

- **Output devices**

a) Monitor display- 13-inch (1440 x 990) Intel HD Graphics 4000 graphics

**Software**

1. **Safari 7.0.6:** last, most updated version of the Safari browser that will be used to test Virtual Queue.
2. **Google Chrome 37.0:** last, most updated version of the Google Chrome browser that will be used to test Virtual Queue.
3. **Spring Framework 4.0:** last, most updated version of Spring Framework, an MVC-based, Java framework used for development. It will be used to develop the front-end and back-end of Virtual Queue.
4. **Spring JDBC template**: make database call access to MySQL database.
5. **MySQL**: database that will be used to store data for the Virtual Queue system.
6. **StarUML:** UML software platform that will be used to create diagrams for the document.
7. **VMWare:** virtual machine software that will be used for the deployment of the software.

## 3.2 Identification of Tasks, Milestones and Deliverables (work breakdown)

Below is a list of all different tasks, milestones, and deliverables for the project, which can be changed according to the development of the project:

|  |  |
| --- | --- |
| Milestones | Task & Deliverables |
| Documentation | * Feasibility Study * Requirement Document * Design Document * Deliverable 4 |
| Environment Setup | * IDE Installation (Eclipse) * JDK 8 Install * StartUML Install * Source Tree for Github * Jboss Installation and Configuration * Maven Project Setup * Spring framework setup * Project dependencies and third parties libraries * Dependency injections setup * MySQL install |
| * Development Tasks: | |
| UI Design | * Create main page template view * Create login page template view * Create My Account page template view * Create Add ride to user page template view * Create Select User template view * Create reset password page template view * Create Add User page template view |
| Database Design and Implementation | * Create Tables * Create tables relationships and constrains * Create EER diagram |
| Main Page Functionality | * Add dynamic content to main page. (JQuery, Ajax calls) * Add server side implementation to main page. * Create main controller with URL definitions and contracts * Create main service interface/implementation * Create main DAO for data access. |
| Login and Logout functionalities | * Add dynamic content to login page (JQuery, Ajax calls) * Server Side Implementation. * Create login controller with URL definitions and UI contracts * Login controller unit test. * Create user Bean. * Unit test to user bean. * Add user information to session when user log * Create login service interface/Implementation * Login service unit test * Create login DAO to facilitate data access * DAO unit test. * Add logout controller with URL contract * Logout controller unit test. * Remove user information from session data. * Unit test. |
| Add User Functionality | * Add to add user page (JQuery, Ajax calls) * Server Side Implementation. * Create User controller with URL definitions and UI contracts * User controller unit test. * Add User information to session when user log * Create User service interface/Implementation * User service unit test * Create User DAO to facilitate data access * DAO unit test. |
| Reset Password Functionality | * Add dynamic content to user page to reset password (JQuery, Ajax calls) * Server Side Implementation. * Add method User controller with URL definitions and UI contracts for password reset. * User controller unit test. * Add functionality to User service interface/Implementation * User service unit test * Add functionality to User DAO to facilitate data access * DAO unit test. |
| User Ride Functionality | * Add dynamic content to user Ride page (JQuery, Ajax calls). * Add UI data validations. * Server Side Implementation. * Create login controller with URL definitions and UI contracts * Add controller unit test * Create Ride Bean. * Unit test to ride bean. * Add user information to session when user log * Create user ride interface/Implementation * Unit test for service interfaces * Create User Ride DAO to facilitate data access * DAO unit test. * Add User Validator interface when registering for a ride. * Validator unit test. * Add User validator implementation with rule definitions. * Validator implementation unit test for each method. * =================================== * Add Ride Registration Business Rules Interface * Add Ride Registration Business Rule implementation. * Add Unit test for Ride Registration * Add Rules Interfaces * Add Rules Implementations. * Add unit test to Rules implementations. * Add Rule Builder Class. * Unit test to Rule Builder. |

## 3.3 Cost of the Project

The cost estimate for each activity will be calculated taking into consideration all the work required to complete the whole project. We will look at human resources and non-human resources costs. The human resources cost will be calculated by the total hours it will be spent to develop the whole project through the entire semester. The non-human resources will be calculated taking into account all hardware and software used in order to develop and complete the whole project.

|  |  |
| --- | --- |
| **Human resource** | **Work (hours)** |
| **Kely Cid** | 520 (13 weeks x 8 daily hors) |

|  |  |  |
| --- | --- | --- |
| **Non-human Resources** | **Quantity** | **Cost** |
| **PC (Hardware)** | 1 | $0.00 |
| **MYSQL** | 1 | $0.00 |
| **Development** |  | $0.00 |
| **Testing** |  | $0.00 |
| **Total Costs** |  | $0.00 |

# 4. Proposed System Requirements

The proposed system is called Virtual Queue, which will give users the functionality to sign in/out of the system of different rides/events, see all the rides they signed on for and received notifications. This chapter will include the functional and non-functional requirements of the system and the requirements analysis phase of the system.

## 4.1 Functional and Non-Functional Requirements

The system shall successfully allow guest users to register.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successful allow users to validate their account.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successful allow venue admin to disable registered user accounts.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successful allow registered users to sign in to their accounts.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successfully reset user’s password upon user request and after confirming his/her identity.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain

The system shall successful allow registered users to logout of their account.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successful allow registered users to have access to the available rides/events.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successful allow registered users to select their favorite rides/events.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successful allow registered users to be added to their selected rides/events.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successful allow registered users to see all the rides/events they signed on for.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successful allow registered users to delete themselves from registered rides/events.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successful allow adding and/or removing registered users from their selected rides according to the queuing algorithm.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successful allow to store and retrieve information regarding to the rides/events.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successful allow to check for duplicates registration or multiple registrations for same user.

* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

The system shall successful allow to handle network connectivity issues.

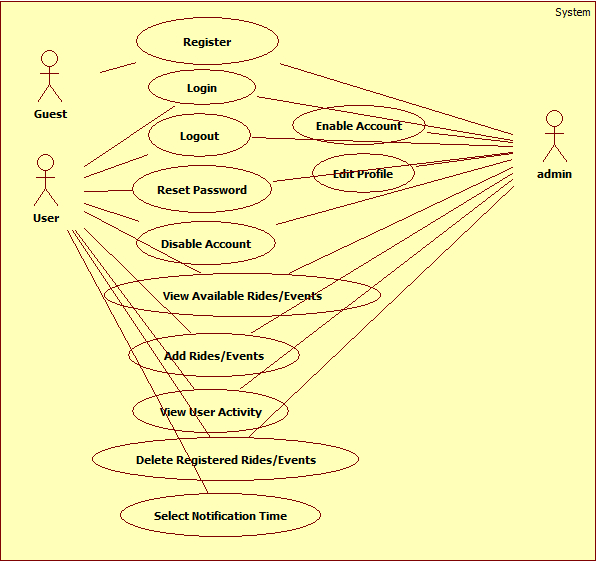
* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

## 4.2 Analysis of System Requirements

This section includes subsections that present the use case model of the VQ system, the static model, and the dynamic model consecutively.

### 4.2.1 Use case model

The use case diagram describes the list of steps that defines the interaction between the three types of users displayed in the diagram: venue guest user, venue registered user, and single venue admin and the system. They all work together to accomplish the goal of the proposed system. Below is the Use Cases Diagrams using UML for specific details.



### 4.2.3 Static model

A static model expresses the system and does not account for sequence of events or time. For the VQ system, a class diagram will be included. The diagram will display the structure of system by showing the classes, attributes, methods, and the relationship between these classes. On Appendix D, the static diagram will be shown.

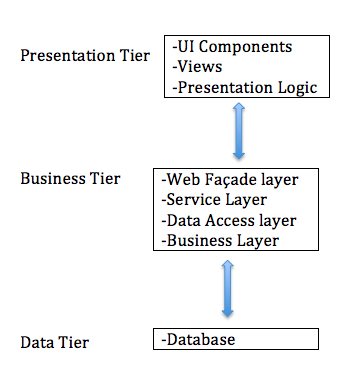
### 4.2.4 Dynamic model

On the other hand, the dynamic model does account for time. For the VQ system, sequence diagrams will be included. These will show objects and class interactions in a sequence of events arranged in a time line that displays functionality in order to allow the developers and programmers to view how the users should transition based on these actions. On Appendix D, the dynamic diagram will be shown.

# 5. System Design (i.e., overall system design)

This chapter will describe the system and subsystem design. It will explain the decomposition of the VQ by giving an overview of the system design architecture. It will provide a detailed description of the subsystem decomposition for each major subsystem. It will cover how the hardware and software are mapped. It will identify the persistent data management that needs to be stored and the structure of the data. Lastly, it will describe security and privacy user authentication processes, encryption of data and all other security parameters being implemented.

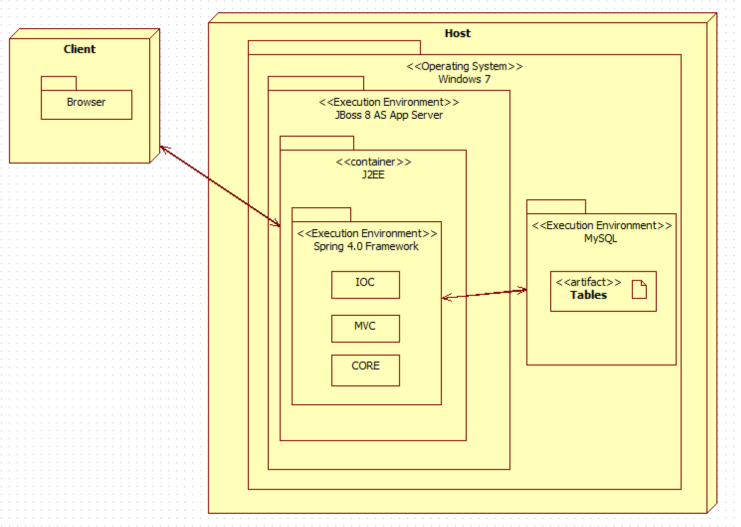
## 5.1 Overview



## 5.2 Subsystem Decomposition

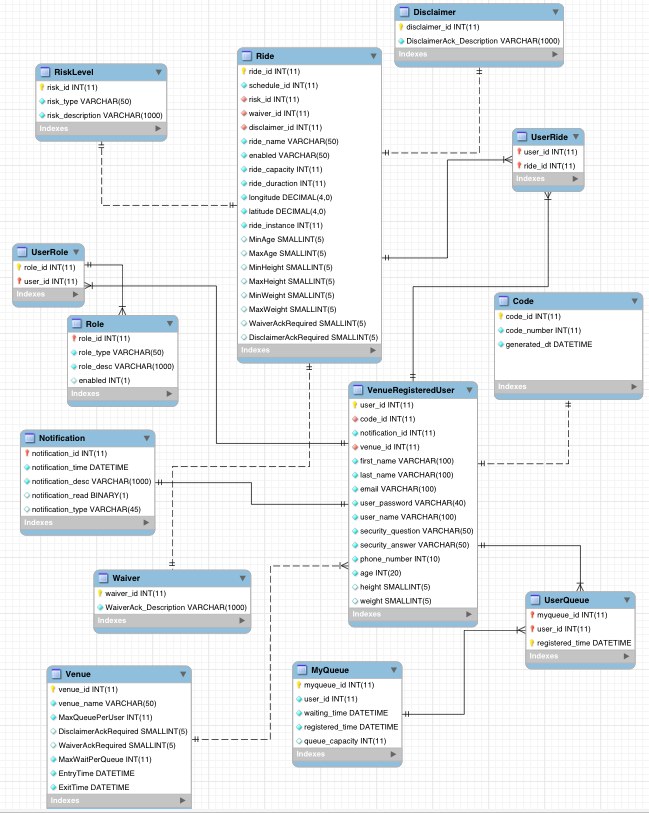
## 5.3 Hardware-Software Mapping

To map the hardware and software used for the Virtual Queue system, a deployment diagram was constructed. It is being hosted on a virtual machine with windows 7 running with JBoss 8 application server and Spring 4.0 framework and MySQL database on the FIU SCIS network. Below is the representation of it.



## 5.4 Persistent Data Management

For the VQ system a database was implemented from scratch, taking into account all the information needed to be stored and support the each functionality. Below is the EER diagram schema with all the tables and relations.



## 5.5 Security/Privacy

For the VQ system all data is password protected. Authentication of each registered user and administrators will be done by checking the database for a registered email, which will be the username as well; also, the table Role and User Role will be checked as well to determine user privileges. Registered users will only have access to their information, and their own data. On the other hand, administrators will have access to all registered users information and results.

# 6. Detailed Design

The detailed design of the VQ system is considered an abstract that translate to source code in a clear way. That is one of the reasons that it has to be detailed and clear. The detailed design chapter will present the system design in a variety of views where each uses a variety of modeling techniques. This chapter is composed for four sections. Section 3.1 is the overview of the chapter; it will give a brief description of the behavior and structure of each subsystem. Section 3.2 contains the static models of each subsystem, and Section 3.3 the dynamic model, which will provide the different diagrams for each subsystem. Section 3.4 delivers the class interfaces and constraints for the main control object in each system.

## 6.1 Overview

## 6.2 Static models

## 6.3 Dynamic Model

## 

## 6.4 Code Specification

# 7. System Validation

The detailed design of the VQ system is considered an abstract that translate to source code in a clear way. That is one of the reasons that it has to be detailed and clear. The detailed design chapter will present the system design in a variety of views where each uses a variety of modeling techniques. This chapter is composed for four sections. Section 3.1 is the overview of the chapter; it will give a brief description of the behavior and structure of each subsystem. Section 3.2 contains the static models of each subsystem, and Section 3.3 the dynamic model, which will provide the different diagrams for each subsystem. Section 3.4 delivers the class interfaces and constraints for the main control object in each system.

## 7.1 Subsystem Tests

## 7.2 System Tests

|  |  |
| --- | --- |
| Test Case ID | VQ01\_Login\_SunnyTest01 |
| Purpose | Test for successful connection to the VQ system with valid username and password. |
| Test Setup | MySQl database must have been setup correctly and program must be running. |
| Input | username: “kcid001@fiu.edu” password: “kely” |
| Expected Results | The user should be granted access to the system and be presented with account page. |

|  |  |
| --- | --- |
| Test Case ID | VQ01\_Login\_SunnyTest02 |
| Purpose | Test for successful connection to the VQ system with valid username and password. |
| Test Setup | MySQl database must have been setup correctly and program must be running. |
| Input | username: “felineyes151@gmail.com” password: “kely1” |
| Expected Results | The user should be granted access to the system and be presented with account page. |

|  |  |
| --- | --- |
| Test Case ID | VQ01\_Login\_RainyTest03 |
| Purpose | Test for connection to the VQ system with valid username and incorrect password. |
| Test Setup | MySQl database must have been setup correctly and program must be running. |
| Input | username: “kcid001@fiu.edu” password: “kely1” |
| Expected Results | The system does not allow access to user. |

|  |  |
| --- | --- |
| Test Case ID | VQ02\_Logout\_SunnyTest04 |
| Purpose | Test for successful logout from the VQ system by logged user. |
| Test Setup | MySQl database must have been setup correctly, program must be running, and user must be logged in. |
| Input | Click the logout button |
| Expected Results | The user is logged out of the VQ system and presented with the Home page. |

|  |  |
| --- | --- |
| Test Case ID | VQ02\_Logout\_SunnyTest05 |
| Purpose | Test for successful logout from the VQ system by logged user. |
| Test Setup | MySQl database must have been setup correctly, program must be running, and user must be logged in. |
| Input | Click the “X” in the top right corner |
| Expected Results | The user is logged out of the VQ system. |

|  |  |
| --- | --- |
| Test Case ID | VQ02\_Logout\_RainyTest06 |
| Purpose | Test for successful logout from the VQ system by logged user. |
| Test Setup | MySQl database must have been setup correctly, program must be running, and user must be logged in. System was unexpectedly disconnected from the database |
| Input | User tried to logout with a disconnected database. |
| Expected Results | Unable to logout dialog. |

|  |  |
| --- | --- |
| Test Case ID | VQ03\_ResetPass\_SunnyTest07 |
| Purpose | Test for successfully resetting password in the VQ system by registered user. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user must be registered and must be on Login menu. |
| Input | User clicks the Reset Password button, fills out the form presented, clicks submit, and try to log back in to the system. |
| Expected Results | User is granted access to the system with the new password. |

|  |  |
| --- | --- |
| Test Case ID | VQ03\_ResetPass\_SunnyTest08 |
| Purpose | Test for successfully resetting password in the VQ system by registered user. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user must be registered and must be on MyAccount menu. |
| Input | User clicks the Reset Password button, fills out the form presented, clicks submit, and try to log back in to the system. |
| Expected Results | User is granted access to the system with the new password. |

|  |  |
| --- | --- |
| Test Case ID | VQ03\_ResetPass\_RainyTest09 |
| Purpose | Test for resetting password in the VQ system by registered user with incorrect security answer. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user must be registered and must be on Reset password form. |
| Input | For Username: [test@test.com](mailto:test@test.com), Security answer: two  User fills out the rest of the reset password form presented correctly, and clicks submit. |
| Expected Results | User is requested to enter the correct security answer on the form. |

|  |  |
| --- | --- |
| Test Case ID | VQ04\_ConfirmIdent\_SunnyTest10 |
| Purpose | Test if correct user by resetting password with valid username and security question. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user must be registered and forgot password. |
| Input | User clicks on Reset Password from Login Page. User enters Username: [test@test.com](mailto:test@test.com), Security answer: one, and fills out the rest of the reset password form presented correctly, and clicks submit. |
| Expected Results | User is validated by the system, and password is correctly changed on the VQ database system. |

|  |  |
| --- | --- |
| Test Case ID | VQ04\_ConfirmIdent\_SunnyTest11 |
| Purpose | Test if correct user by resetting password with valid username and security question. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user must be registered and must want to reset password. |
| Input | User clicks on Reset Password from Account Page. User enters Username: [test@test.com](mailto:test@test.com), Security answer: one, and fills out the rest of the reset password form presented correctly, and clicks submit. |
| Expected Results | User is validated by the system, and password is correctly changed on the VQ database system. |

|  |  |
| --- | --- |
| Test Case ID | VQ04\_ConfirmIdent\_SunnyTest12 |
| Purpose | Test if correct user by resetting password with wrong username and security question. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user must be registered and must want to reset password. |
| Input | User clicks on Reset Pasword from Login Page. User enters Username: [test11@test.com](mailto:test11@test.com), Security answer: wrong, and fills out the rest of the reset password form presented correctly, and clicks submit. |
| Expected Results | User is validated and asked to correct the reset password form. Password is not changed on the VQ database system. |

|  |  |
| --- | --- |
| Test Case ID | VQ05\_RegisterNewUser\_SunnyTest13 |
| Purpose | Test to see if VQ system correctly adds a user to the VQ database. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user requires an account. |
| Input | User clicks on Register button from the Home page, user fills the register form presented correctly, and clicks submit. |
| Expected Results | User is able to log in to the VQ system with the credential chosen. |

|  |  |
| --- | --- |
| Test Case ID | VQ05\_RegisterNewUser\_SunnyTest14 |
| Purpose | Test to see if VQ system correctly adds a user to the VQ database. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user tries to login to the system without an account. |
| Input | User clicks on Register button from the login page, user fills the register form presented correctly, and clicks submit. |
| Expected Results | User is able to log in to the VQ system with the credential chosen. |

|  |  |
| --- | --- |
| Test Case ID | VQ05\_RegisterNewUser\_RainyTest15 |
| Purpose | Test to see if VQ system adds a user to the VQ database that already exists. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user chooses to enter an already registered account. |
| Input | User clicks on Register button from Home page, fills the register form presented with username: [test@test.com](mailto:test@test.com) already on database, and clicks submit. |
| Expected Results | User is requested to choose another username because that one is already been used. |

|  |  |
| --- | --- |
| Test Case ID | VQ06\_SelectRide\_SunnyTest16 |
| Purpose | Test if user can select a ride that is available. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user is logged in. Rides are made available. |
| Input | User clicks on Rides button from the Home page, clicks the dropdown menu presented and selects the desired available ride. |
| Expected Results | VQ system should allow the user to select the desired ride. |

|  |  |
| --- | --- |
| Test Case ID | VQ06\_SelectRide\_SunnyTest17 |
| Purpose | Test if user can select a ride that is available. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user is logged in. The VQ system has just 3 rides available. |
| Input | User clicks on Rides button from the Home page, clicks the dropdown menu presented and selects one the three available rides. |
| Expected Results | VQ system should allow the user to select the desired ride. |

|  |  |
| --- | --- |
| Test Case ID | VQ06\_SelectRide\_RainyTest18 |
| Purpose | Test if user can select a ride that is available. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user is logged in. Rides are all unavailable. |
| Input | User clicks on Rides button from the Home page, clicks the dropdown menu presented and tries to select a ride. |
| Expected Results | VQ system should not allow the user to select any ride. |

|  |  |
| --- | --- |
| Test Case ID | VQ12\_ViewRides\_SunnyTest34 |
| Purpose | Test if users can see the available rides. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user is logged in. Rides are made available. |
| Input | User clicks on Rides button from the Home page, clicks the dropdown menu presented. |
| Expected Results | User can see the rides available at the venue. |

|  |  |
| --- | --- |
| Test Case ID | VQ12\_ViewRides\_SunnyTest35 |
| Purpose | Test if users can see the available rides. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user is logged in. Some Rides are made available, and others unavailable. |
| Input | User clicks on Rides button from the Home page, clicks the dropdown menu presented. |
| Expected Results | User can see the updated rides available at the venue. |

|  |  |
| --- | --- |
| Test Case ID | VQ12\_ViewRides\_RainyTest36 |
| Purpose | Test if users can see the available rides. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user is logged in. Rides are all unavailable. |
| Input | User clicks on Rides button from the Home page, clicks the dropdown menu presented. |
| Expected Results | User cannot see any available rides at the venue. |

|  |  |
| --- | --- |
| Test Case ID | VQ12\_ViewActivity\_SunnyTest37 |
| Purpose | Test if users can see their account activity. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user is logged in and on his/her account page. User has not added any rides yet. |
| Input | No input necessary. |
| Expected Results | User can see his/her account page with no rides added. |

|  |  |
| --- | --- |
| Test Case ID | VQ12\_ViewActivity\_SunnyTest38 |
| Purpose | Test if users can see their updated account activity. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user is logged in and on his/her account page. User has been added to at least 1 ride. |
| Input | No input necessary. |
| Expected Results | User can see his/her account page with the ride that was added. |

|  |  |
| --- | --- |
| Test Case ID | VQ12\_ViewActivity\_RainyTest39 |
| Purpose | Test if users can see their account activity. |
| Test Setup | MySQl database must have been setup correctly, program must be running, user is logged in and on his/her account page. User has canceled all rides. |
| Input | No input necessary. |
| Expected Results | User can see his/her account page with no rides. |

## 7.3 Evaluation of Tests

On the table below are each of the system test cases id for the VQ system that were conducted and the actual results.

|  |  |
| --- | --- |
| Test Case ID | Actual Test Results |
| VQ01\_Login\_SunnyTest01 | PASS |
| VQ01\_Login\_SunnyTest02 | PASS |
| VQ01\_Login\_RainyTest03 | PASS |
| VQ02\_Logout\_SunnyTest04 | PASS |
| VQ02\_Logout\_SunnyTest05 | PASS |
| VQ02\_Logout\_RainyTest06 | PASS |
| VQ03\_ResetPass\_SunnyTest07 | PASS |
| VQ03\_ResetPass\_SunnyTest08 | PASS |
| VQ03\_ResetPass\_RainyTest09 | PASS |
| VQ04\_ConfirmIdent\_SunnyTest10 | PASS |
| VQ04\_ConfirmIdent\_SunnyTest11 | PASS |
| VQ04\_ConfirmIdent\_RainyTest12 | PASS |
| VQ05\_RegisterNewUser\_SunnyTest13 | PASS |
| VQ05\_RegisterNewUser\_SunnyTest14 | PASS |
| VQ05\_RegisterNewUser\_RainyTest15 | PASS |
| VQ06\_SelectRide\_SunnyTest16 | PASS |
| VQ06\_SelectRide\_SunnyTest17 | PASS |
| VQ06\_SelectRide\_RainyTest18 | PASS |
| VQ12\_ViewRides\_SunnyTest34 | PASS |
| VQ12\_ViewRides\_SunnyTest35 | PASS |
| VQ12\_ViewRides\_RainyTest36 | PASS |
| VQ12\_ViewActivity\_SunnyTest37 | PASS |
| VQ12\_ViewActivity\_SunnyTest38 | PASS |
| VQ12\_ViewActivity\_RainyTest39 | PASS |

# 8. Glossary

**Actor:** Anyone that can use or misuse the system.

**Database:** A collection of related data.

**Deliverable:** A work product for the client.

**Expected Results:** The results that will be produced when executing a test case if and only if the program satisfies its intended behavior.

**Role:** Associated with a set of tasks assigned to a project participant.

**Test Setup:** A status that must be true before an operation is invoked.

**Stub:** A skeletal or special-purpose implementation of a software module, used to develop or test a module that calls or is otherwise dependent on it.

**System:** Underlying reality.

**System Testing:** Tests all the components together, seen as a single system to identify errors with respect to the scenarios from the problem statement and the goals in the goals in the requirements specification and system design.

**Task:** An atomic unit of work that is both manageable and consumes resources.

**Test Case:** A set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement.

**Test Driver:** A software module used to invoke a module under test and, often, provide test inputs, control and monitor execution, and report test results.

**Input:** The input necessary to complete some execution of the software under test.

**Use Case:** A general sequence of events that defines all possible actions between one or many actors and the system for a given piece of functionality.

**User Interface (UI):** The way through which a user interacts with the computer system.

**Class Diagram:** A pictorial representation of all the classes in the system

**Functional Requirement:** A function supported by the system, where a function is a set of inputs, the behavior, and outputs.

**Non-Functional Requirement:** A requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors.

**Object Diagram:** A pictorial representation of an instance of a class with example of how the data of the class will be populated.

**Sequence Diagram:** A pictorial representation of how processes operate with one another and the user during the course of a specific piece of functionality.

# 9. Appendix

This chapter contains a Gantt Chart with the scheduled time of work for the whole project, all use cases with nonfunctional requirements, the user interface design of the VQ system, static and dynamic diagrams for the analysis model and design models as well, the documented class interfaces and constrains, the documented code for test drivers and stubs, and a diary of meetings for the entire semester.

## 9.1 Appendix A - Project schedule (Gantt chart or PERT Chart)

## 9.2. Appendix B – All use cases with nonfunctional requirements

## 9.3. Appendix C – User Interface Designs

## 9.4. Appendix D – Analysis models (static and dynamic)

## 9.5. Appendix E – Design models (static and dynamic)

## 9.6. Appendix F – Documented class interfaces (code) and constrains

## 9.7. Appendix G – Documented code for test drivers and stubs

## 9.8. Appendix H – Diary of Meetings and Tasks for the entire semester

**Virtual Meeting 1:**

**Date:** September 2, 2014

**Start Time:** 9:16PM

**Participating members:** Kely Cid, Bernard Parenteau.

**Agenda:** Programming Background discussion.

**Virtual Meeting 2:**

**Date:** September 3, 2014

**Start Time:** 12:45PM

**Participating members:** Kely Cid, Bernard Parenteau.

**Agenda:** Agreed on a optimized website development for the Virtual Queue System, and discussed Virtual Queue system requirements.

**Virtual Meeting 3:**

**Date:** September 3, 2014

**Start Time:** 3:48PM

**Participating members:** Kely Cid, Bernard Parenteau.

**Agenda:** Agreed to go forward with project requirements, technologies and tools.

**Meeting 4:**

**Date:** September 3, 2014

**Start Time:** 6:20PM

**End Time:** 8:35PM

**Participating members:** Kely Cid.

**Agenda:** Understanding of the project objectives and goals, and started working on Chapter 1 & 2 of the feasibility and project plan document.

**Meeting 5:**

**Date:** September 4, 2014

**Start Time:** 2:10PM

**End Time:** 5:30PM

**Participating members:** Kely Cid.

**Agenda:** Continue working on Chapter 2 of the feasibility and project plan document, and chapter 1 of requirement document.

**Meeting 6:**

**Date:** September 5, 2014

**Start Time:** 2:00PM

**End Time:** 6:00PM

**Participating members:** Kely Cid.

**Agenda:** Continue working on Chapter 2 & 3 of the requirement document, and chapter 3 of feasibility and project plan document. Created Trello Account.

**Meeting 7:**

**Date:** September 6, 2014

**Start Time:** 1:00PM

**End Time:** 7:10PM

**Participating members:** Kely Cid.

**Agenda:** Continue working on Chapter 3, 4, 5, and part of 6 of the requirement document.

**Meeting 8:**

**Date:** September 7, 2014

**Start Time:** 11:00AM

**End Time:** 12:45PM

**Participating members:** Kely Cid.

**Agenda:** Update Trello Account, create Power Point presentation.

**Meeting 9:**

**Date**: September 14, 2014

**Start Time**: 10:30AM

**End Time**: 7:50PM

**Participating members**: Kely Cid, Bernard Parenteau.

**Agenda**: Discussed and Worked on UI design. Update mentor on project. Uploaded mockups for approval on Trello account of Login, Select Ride, and Reset Password and received feedback for mockups.

**Meeting 10:**

**Date**: September 16, 2014

**Start Time**: 10:30AM

**End Time**: 7:50PM

**Participating members**: Kely Cid, Bernard Parenteau.

**Agenda**: Update mentor on project. Define views, controllers, models and url end point contracts for server side end points and test it with static data before connecting UI with server side controller. Database design and implementation.

**Meeting 11:**

**Date**: September 22, 2014

**Start Time:** 1:00PM

**End Time**: 10:45PM

**Participating members**: Kely Cid.

**Agenda**: Keep working on UI design. Added dynamic features to pages. Created packages to group all controllers, services, and DAO's. Services and DAOs interfaces and implementation for each one were created. Posted half of use cases on Trello for approval. Received approval of Use case diagram V2 for VQ system.Uploaded code to GitHub account.

**Meeting 12:**

**Date**: September 23, 2014

**Start Time:** 1:00PM

**End Time**: 10:45PM

**Participating members**: Kely Cid, Bernard Parenteau.

**Agenda**: Update mentor on project. Discuss Mockups design and use cases for VQ system were uploaded to Trello.

**Meeting 13:**

**Date**: September 25, 2014

**Start Time**: 10:30AM

**End Time**: 7:50PM

**Participating members**: Kely Cid, Bernard Parenteau.

**Agenda**: Update mentor on changes for Feasibility document and feasibility matrix and updates on provided VM with all tools needed already installed.

**Meeting 14:**

**Date**: September 30, 2014

**Start Time**: 10:30AM

**End Time**: 7:50PM

**Participating members**: Kely Cid, Bernard Parenteau.

**Agenda**: Approval received for Mockups designs.

**Meeting 15:**

**Date**: October 2, 2014

**Start Time**: 10:00AM

**End Time**: 8:00PM

**Participating members**: Kely Cid, Bernard Parenteau.

**Agenda**: Discussed Register and Login functionalities and started working on it to finish it. Started working on Design document and discuss and asked for approval of use cases. Updated Requirement document and sequence diagrams.

**Meeting 16:**

**Date**: October 7, 2014

**Start Time**: 9:30AM

**End Time**: 5:30PM

**Participating members**: Kely Cid, Bernard Parenteau.

**Agenda**: Preliminary UI design approved on Trello.

**Meeting 17:**

**Date**: October 10, 2014

**Start Time**: 9:30AM

**End Time**: 5:30PM

**Participating members**: Kely Cid, Bernard Parenteau.

**Agenda**: Discuss and understand the process of Selecting and Adding a Ride by a user functionalities and all validations to take into account for ride and each user.

**Meeting 18:**

**Date**: October 15, 2014

**Start Time**: 11:00AM

**End Time**: 6:25PM

**Participating members**: Kely Cid.

**Agenda**: Continue working on the Feasibility document, Requirement document, and Design document and functionalities of the system. Update mentor on project.

**Meeting 19:**

**Date**: October 20, 2014

**Start Time**: 9:30AM

**End Time**: 5:30PM

**Participating members**: Kely Cid, Bernard Parenteau.

**Agenda**: Update mentor on project. Continue working on the Feasibility document, Requirement document, and Design document to finish them. Worked on code implementation for logout and reset password and validation rules for adding a ride by a venue user. Added the rest of the use cases to Trello for approval.

# 10. References

Images used:

1. <http://www.plan-family-reunions.com/themeParks.html>
2. http://ru.forwallpaper.com/wallpaper/abstract-circles-patterns-dots-light-colors-bokeh-abstraction-172728.html