**Requirement Document**

CIS 4911 – Senior Project

Virtual Queue

**Member:**

Kenneth Kon

Michael Lazo

**Instructor**

Masoud Sadjadi

**Mentor**

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

**Date**

February 13th 2015

# 1.Proposed System Requirements

The proposed system is called Virtual Queue, which will give users the functionality to sign in and see all rides available at the venue and their respective waiting time. Users will also be able to add any ride they want as long as there is no time conflict between them. They can also delete any ride from their account of they decide to do so, if not, they will receive a notification before their time for that approaches and they will automatically be dequeue from their rides once their time for that ride comes up. Also the user admin will be able to login and edit an user info as well as enable and disable user’s account. This chapter will include the functional and non-functional requirements of the system and the requirements analysis phase of the system.

## 1.1 Functional and Non-Functional Requirements

Allow unregistered 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.

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.

Allow registered users to delete 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.

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.

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.

Allow users to reset their password.

* **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.

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 less than a second.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

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 less than a second.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

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 less than a second.
* **Supportability**: The system should be easy to maintain, make appropriate changes, and be correctly handled by IE, Mozilla, Chrome and Safari.

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.

Allow the user to be notified before their rides/events time occurs.

* **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.

Allow the system to add and/or remove registered users from their selected rides.

* **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.

Allow the system to send notification to registered user before ride time added approaches.

* **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.

Allow the system 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.

Allow the system 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.

Allow the system to handle network connectivity issues.

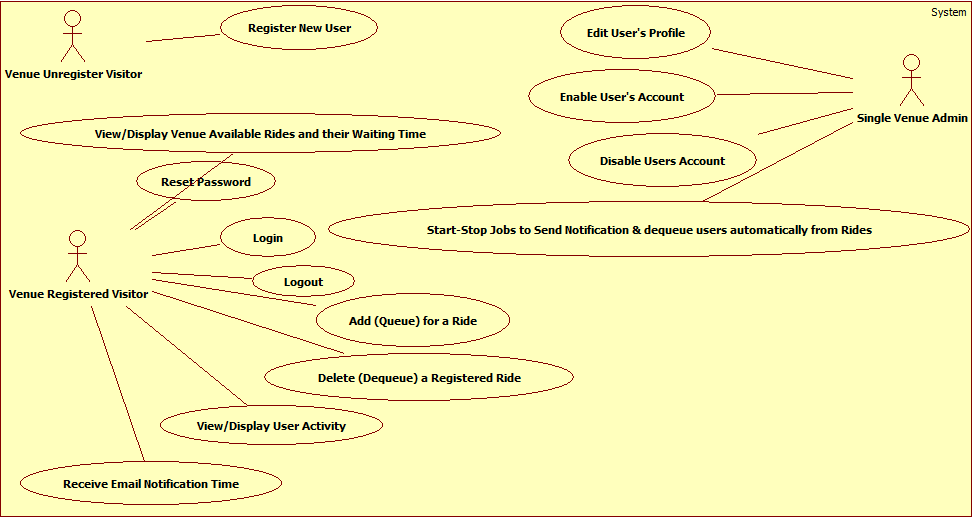
* **Usability**: No previous training time. System is simple and easy to follow.
* **Reliability**: The system should perform correctly 99% of the time.
* **Performance**: The system should be sent and saved within 3 seconds.
* **Supportability**: The system should be easy to maintain, make appropriate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

## 1.2 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.

### 1.2.1 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 with the specific functionalities that were implemented.



### 1.2.3 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.

### 1.2.4 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.