3. Specific Requirements

3.1 Functional Requirements

3.1.1 Graphical User Interfaces

3.1.1.1 Welcome

3.1.1.2 User Settings Page

3.1.1.2.1 Provide the capability to allow users to save parking space type and parking permit preference. (Michael, Imani)

3.1.1.2.2 Provide a mechanism for users to import courses. (Michael, Imani)

3.1.1.2.3 Provide a mechanism for users to import tentative schedules. (Michael, Imani)

3.1.1.3 Garages

3.1.1.3.1 Provide the capability to view a floor plan for a selected garage that displays exactly which spaces are available on each floor. (Gerard)

3.1.1.3.2 Provide capability to show past occupancy for a particular parking garage. (Michael, Cody)

3.1.1.4 Search

3.1.1.4.1 Provide users with the capability of receiving parking suggestions based on imported schedules. (Matt, Sangeet)

3.1.1.4.2 Provide the capability to conduct an advanced search for available parking spaces based on a Starting address. (Gerard)

3.1.1.4.3 Provide the capability to conduct an advanced search for available parking spaces based on Permit Type(s). (Gerard)

3.1.1.4.4.Provide the capability to conduct an advanced search for available parking spaces based on a Space Type(s). (Gerard)

3.1.1.4.5 Provide the capability to conduct an advanced search for available parking spaces based on a Destination building. (Gerard)

3.1.1.4.6 When executing an advanced query, users must be allowed to specify their desired minimum number of available spaces that a garage must have in order to return search results. (Matt, Sangeet)

3.1.1.5 Results

3.1.1.5.1 When returning results from an advanced query, ParkODU must provide the capability to display the directions to each garage from the user's current location utilizing the Google Maps API. (Matt, Sangeet)

3.1.1.6 User Event Notification

3.1.1.6.1 Provide the capability for the user to view event notification as a pop up on the main page of the application. (Matt, Sangeet)

3.1.1.7 Administration Tools

3.1.1.7.1 **Notifications**

3.1.1.7.1.1 Provide the capability for an administrative user to add event notifications. (Ahsif, Isaac)

3.1.1.7.1.2 Provide the capability for an administrative user to edit event notifications. (Ahsif, Isaac)

3.1.1.7.1.3 Provide the capability for an administrative user to delete event notifications. (Ahsif, Isaac)

3.1.1.7.2 **Garage**

3.1.1.7.2.1 Provide the capability for an administrative user to add garages. (Gerard)

3.1.1.7.2.2 Provide the capability for an administrative user to edit garages. (Michael, Cody)

3.1.1.7.2.3 Provide the capability for an administrative user to deleted garages. (Michael, Cody)

3.1.1.7.2.4 Provide the capability for an administrative user to add floors. (Gerard)

3.1.1.7.2.5 Provide the capability for an administrative user to edit floors. (Michael, Cody)

3.1.1.7.2.6 Provide the capability for an administrative user to deleted floors. (Michael, Cody)

3.1.1.7.2.7 Provide the capability for an administrative user to add spaces. (Gerard)

3.1.1.7.2.8 Provide the capability for an administrative user to edit spaces. (Michael, Cody)

3.1.1.7.2.9 Provide the capability for an administrative user to deleted spaces. (Michael, Cody)

3.1.1.7.3 **Events**

3.1.1.7.3.1 Provide the capability for an administrative user to schedule events that affect the status of a garage. (Ahsif, Isaac)

3.1.1.7.3.2 ParkODU provides administrative users with a service that executes scheduled tasks created by an administrator. (Ahsif, Isaac)

3.1.1.7.4 **Account Management**

3.1.1.7.4.1 Provide the capability for an administrative user to add other administrative accounts. (Matt, Imani)

3.1.1.7.4.2 Provide the capability for an administrative user to edit other administrative accounts. (Matt, Imani)

3.1.1.7.4.3 Provide the capability for an administrative user to delete other administrative accounts. (Matt, Imani)

3.1.1.8 **Login**

3.1.1.8.1 Provide the capability for all users to login to ParkODU. (Matt, Sangeet)

3.1.1.8.2 Provide the capability for all users to log out of ParkODU. (Matt, Sangeet)

3.1.2 Application Programming Interface (API)

3.1.2.1 **Garages**

3.1.2.1.1 Provide the capability for users and external applications to query for garage status. (Gerard)

3.1.2.1.2 Provide the capability for users and external applications to query for garage count. (Gerard)

3.1.2.1.2 Provide the capability for users and external applications to query for garage floor count. (Gerard)

3.2 Non-functional

3.2.1 Reliability

3.2.1.1 Provide high availability access.

3.2.2 Security

3.2.2.1 Establish a user access list.

3.2.2.2 Integrate a source code vulnerability tool during the development process.

3.2.3 Maintainability

3.2.3.1 Define standard operating procedures.

3.3 Assumptions and Constraints

3.3.1 Assumptions

It is assumed that the customer will have a parking garage. It is also assumed that the customer employs staff for maintaining the parking garages and their web hosting services. Final product (not prototype) requires a vehicle detection technology.

3.3.2 Constraints

Typically vehicle detection technology would be in place, however for the prototype a simulation of parking conditions will be used.

[This space intentionally left blank]

Appendix

Required Equipment

- Linux Servers

Required Software

- Java 8+ JDK
- MongoDB
- Gradle
- Git
- IntelliJ Community Edition
- Hazelcast