

**Department of Transportation**

**Requirements Documentation**

**The Honey Badgers:**

*Marcus Shannon, Will Hicks, Adam Mitchell,   
Jarred Wynan, and Kyle Pirnie*

**Abstract:**

The department of transportation associated with the University of Louisiana at Lafayette has requested a software application be developed. Overall, what has been requested is a smartphone application that will inform users about how many parking spaces are available inside any specifically given parking tower. This application should be available for free so any student may download it from the app store to learn more about parking information on campus. It’s also been requested that all development resources be well developed in such a way that a future development team may also iterate and build on to the existing software. This document will outline the use case provided by the client and our formalization of the use case using UML diagrams.

**Table of Contents:**

* **List of Figures**
  + *Figure 1.1 - Overall User Interaction*
  + *Figure 1.2 - User to Mobile Application interaction*
  + *Figure 1.3 - User to Administration Panel interaction*
* **Introduction**
  + *Scope*
  + *Document Structure*
* **Use Case Model for Functional Requirements**
  + *Graphic Use Case Model*
  + *Textual Description*
    - *Specific Use Cases*
* **Rationale for the Use Case Model**
* **Non-Functional Requirements**
* **Evidence the requirements have been placed under configuration management control**

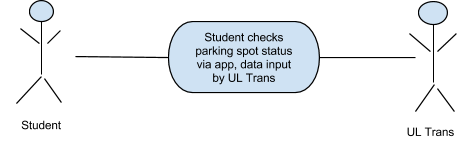
**Introduction:**

As discussed in the introduction for the Project Management Plan documentation, the overarching goal of the software we are developing is to communicate the number of available parking spaces in the parking tower. This document will outline something known as UML use case diagrams. Use case diagrams are highly useful when communicating the structure of the program, or how the program will work, to the client. It also allows for the client to understand all features of the product so no ambiguity may exist and everyone will understand what is desired.

* **Scope:** For the scope of this document, we will only focus on an extremely high level of abstraction to completely understand the focus or end goal for the main product.
* **Document Structure:** This document structured in such a way that someone without a technical background may read and understand the project along with all features required for this project.

**Use case model for Functional Requirements:**

-*Overall User Interaction*

Figure 1.1 

**Description:**

This use case diagram demonstrates the highest level of abstraction for user interaction. The ULL Department of Transportation will either be hiring someone part-time or installing new hardware to input information into the computer system. After this information is input into the computer, anyone that attempts to start their mobile application will automatically connect to this computer system and receive an update about how many parking spaces are available in the referenced parking tower.

-*User to mobile application interaction*

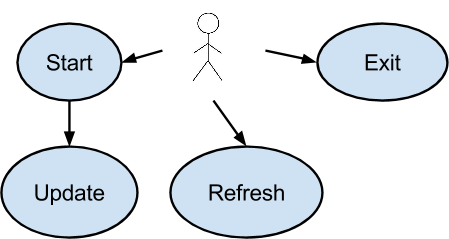


Figure 1.2

**Description:**

Initially, the user will only have 3 types of key functionality for the mobile application. Starting the application which will do an initial update to get the most current information from the computer systems about current parking availability. The second key functionality will be a “Refresh button” which will force a connection with the computer system to get up-to-date information about current parking availability. Lastly the application will be able to exit quickly and elegantly.

-*User to Administration Panel interaction*

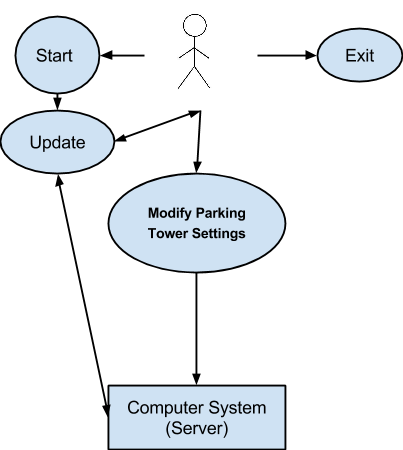


Figure 1.3

**Description:**

Idealistically, the Administration Interface will be used to monitor the actual usage of the mobile application. With the initial start up of the Administration interface, it will receive live-feed updates from the server to gather some statistics on it’s actual usage. An example of this would be how many people have connected to the server in the last hour/day/month/all. The user could also add parking towers, adjust maximum number of parking spots available, adjust current number of parking spots free, any other options deemed relevant. This will be more expanded into detailed on the technical documents. Lastly the application will be able to exit quickly and elegantly.

**Rationale for the Use Case Model**

This particular model is being implemented in order to make use of the information we have[the handheld information from UL Trans] and to most effectively get it into the hands of users. We anticipate that our users will be looking for spots as close to their arrival as possible as to get the most up to date information as possible. Limited resources prevent electronic counters to be installed in the parking structure, so the parking enforcement officers(PEO’s) will input the data to give us a rough count to display to students.

To best serve this need a phone application is appropriate. Driving students will need to make a decision whether to park on campus or avoid doubling back and wasting time in traffic after realizing there are no available spots. While we don’t condone using your phone while driving, we understand that a mobile app would be the most effective and safest way of relaying information . If we had the count displayed on a website, users would most likely have to open up a web browser in their phone, and navigate to the page. This takes time, and if our user is driving, it also takes away their concentration. Thus, a mobile app that displays the count immediately when opened would be safer than a website.

**Non-functional Requirements:**

The UL Transportation Dept is growing along with the rest of the university. As such it is currently evaluating new technologies to help better serve the students, the university and the community. This technology will enhance the level of information available to administrators and students, as well as decrease traffic congestion near the parking structures. The available parking space count should be accurate and as close to a real time count as possible. To avoid human error, an electronic loop or ticket counter should communicate with our app in order to get an accurate count of the spaces available. At the moment, resources don’t allow for these devices, so the PEO(parking enforcement officer) will relay the count.

We do not anticipate the parking machine’s arrival before our project deadline. We are still attempting to account for this incoming tech as much as possible in our project planning by making sure our software is maintainable, understandable(well-commented), and adaptable.

EVIDENCE THE REQUIREMENTS HAVE BEEN PLACED UNDER

CONFIGURATION MANAGEMENT

<https://github.com/somecoder43242/CMPS_453.git>

REFERENCES