First Iteration Requirements

The below features will be developed on the website version of our product, which we have designated the full version. We will also create a bare bones version of our Android App, but we will only implement the login and signup functionality on the Android App.

Task: Creating the database
Responsible: Jessica Yeh
We will reference the database model from the technical design document and write MySQL queries to create the database and the tables that we specified in that model.

Task: Login Feature (Regular)
Responsible for GUI: James Lomeo
Responsible for Database: James Lomeo
The login screen will consist of a regular login as used in Custom Cupcakes. We will implement the Facebook/Google API login at a later iteration. The login screen will take the user to an account page upon a successful login. If the login is unsuccessful the user will receive an error.

Task: Signup Feature (Regular)
Responsible for GUI: James Lomeo
Responsible for Database: James Lomeo
The signup screen will consist of signing up using a standard name, address, email, phone form. In the future, a user will be able to log in with Facebook/Google credentials. For the first iteration, a user will need to create an account with PonyPark before continuing.

Task: View Garages Feature (List View)
Responsible for GUI: Evan Kohn and Justin Trantham
Responsible for Database: Jordan Kayse
The List View will display all of the garages stored within the main database in a row based format. Each row will contain three columns; the most recent rating, the name of the garage, and the distance from the user to the garage. All data is dynamically displayed using JSON objects handed to us by the backend.

The list of garages will be retrieved from a query that will get the name, location and rating of the garages. The location will be used to find the distance from a user. For the first iteration, we will use a random number for the rating, while a future iteration will include the average rating of each garage. We will then return the queries as a JSON file.

Task: View Garages Feature (Map View)
Responsible for GUI: Evan Kohn and Justin Trantham
Responsible for Database: Story Zanetti
The Map View will use the Google Maps API. The default map view will be focused on the SMU campus. Each garage will be shown as a pin, and clicking on a pin will bring up a

dialog box allowing the user to see the garage details (name, address, rating, distance). The dialog box will also contain a Rate button that takes the user to a page where they can report the availability of the selected garage.

Similar to the list view feature, the garage details (name, address, rating, distance) in the dialog box associated with each garage will be retrieved from a query to the database. The results of this query will be returned as a JSON object. The Rate button will insert a user entered rating into the database, as is described in the task below. For the first iteration, the rating displayed will be a random number. At a later iteration, the rating will be an average rating for that garage.

Task: Rate Garages Feature
Responsible for GUI: Evan Kohn
Responsible for Database: Jessica Yeh
The Rate Garages screen will display the name, address, and picture (if one is available) of a particular garage. The screen will have buttons allowing the user to rate a level of the garage on a scale of Full-Empty. For the first iteration, each button will be fully functional and send the user's rating to the database. In a future iteration, we will allow the user to type in an estimate of the number of available spaces.

We will write queries that can handle the rating of garages. For the first iteration, we will be able to take in a user's rating for a particular garage, and insert that into the database. For a future iteration, we will take the ratings and average them so we can display to a user information about how full a parking garage is.

Task: Creating the Android App with Appropriate Features
Responsible: Justin Trantham
For the first iteration, the main goal is to develop a base structure and platform to later build upon. We narrowed down the tasks to the bare bone basics of allowing the users to signup/login and browse blank pages.

The login screen will consist of two text fields (EditText fields) that will represent the username and password fields. The user's data will be retrieved using field.getText(). There will also be a JSONParser class which will then send this information gathered via an HTTP request. The signup screen is very similar to the login screen but will have three text fields for the username, email, and password. It will also utilize the JSONParser class to send/request data via HTTP requests to the remote server.

Finally, we will create a skeleton of the Android application. We will design the structure/organization of the pages listed below as minimally as possible. The navigation will use the Action Bar, the Action Bar will be a mixture of action items, tabs, and icons.

> *Home Screen*, *Report Availability*, *ListView*, *MapView*, *Favorite Garage List*, *My Account*, and *Navigation* (how the user gets from one page to another)