Department of Computer Science and Engineering
The University of Texas at Arlington

Team: Sense

Project: SmartPark

Team Members:

Oreoluwa Adebambo
Victor Diaz
Gaurab Gautam
Braden Walters

Last Updated: July 14, 2014; 1:02 am

Table of Contents

Document Revision History

Revision Number	Revision Date	Description	Rationale
0.1	3 July 14	SRS First Draft	First draft document
0.2	11 July 14	Revise draft based on instructor comments	SRS Gate Review
0.3	14 July 14	Revised SRS based on Breaking Bat's comments	SRS V1.0

List of Figures

List of Tables

1. Product Concept

This section describes the purpose, use and intended user audience for the SmartPark product. SmartPark is an intelligent parking system that is designed to facilitate the localization of parking spots either in a city, university, or parking garages reducing the frustration of looking for parking in busy spaces. This product will save countless hours of civilians' lives spent looking for parking that can be applied to do other things that they would rather be doing. As a result it will lower emissions and gas used by consumers, therefore saving money and time.

1.1 Purpose and Use

The demand for accessible and quick parking has always been one of the top wishes for any average person. While there are several solutions which show at a high level whether the parking lot is full or not, there is a scarcity of a smart solution which can tell exactly which parking spot is open, if any, and directions to it. For the current scope of this project we will not be providing directions.

SmartPark is a system that serves drivers and business owners that are looking for an intelligent parking solution. An average user will be able to find parking spots through a mobile or web application, this application will show by location which parking spaces are open or closed in a parking garage, parking lot, or in the city, to help the user avoid circling around while looking for parking. The SmartPark system will help the user plan his commute and parking ahead of time, saving all of the frustration that is involved when looking for parking. This functionality is helpful when special events, games or fairs occur. Also, it helps in downtown locations or places with which the user is not familiar. This will help our users save time by planning ahead and money by reducing the amount of gas wasted circling around parking lots.

It will allow business owners, especially "Valet Parking" management companies to manage their parking spaces more efficiently and manage their staff better. Retail businesses will be able to generate data on customer volume and related data to help in their decision making process while helping their customers find parking easier.

From a driver's standpoint, the SmartPark system will help them plan ahead for where they need to go and show which parking spots are available at that time in the surrounding areas.

All this information will be presented to the user in a mobile or web application so they can make parking decisions based off of the data we provide them, making their quality of life and commute a little more enjoyable.

1.2 Intended Audience

The intended audience for the SmartPark system includes commuters, business owners, city managers, university campuses, valet parking managers, and airports. For the current scope of work, we are planning to focus on two specific customer categories:

1. Valet Parking Managers
2. University Students

Valet parking managers will be able to use SmartPark to manage their parking spaces more efficiently. They will get a view of what parking spots are available at any given time. They will provide their staff with a mobile application to show the available spaces in that parking lot so the staff can efficiently get to that parking spot in the least amount of time and ready to serve the next valet customer.

University students will expect to be able to open their mobile application and verify if there is parking available in a lot and if there is the application should tell them where. If the student has given us permission to access their calendar, we will notify them of available parking spaces when their scheduled time approaches or they get close to their desired location.

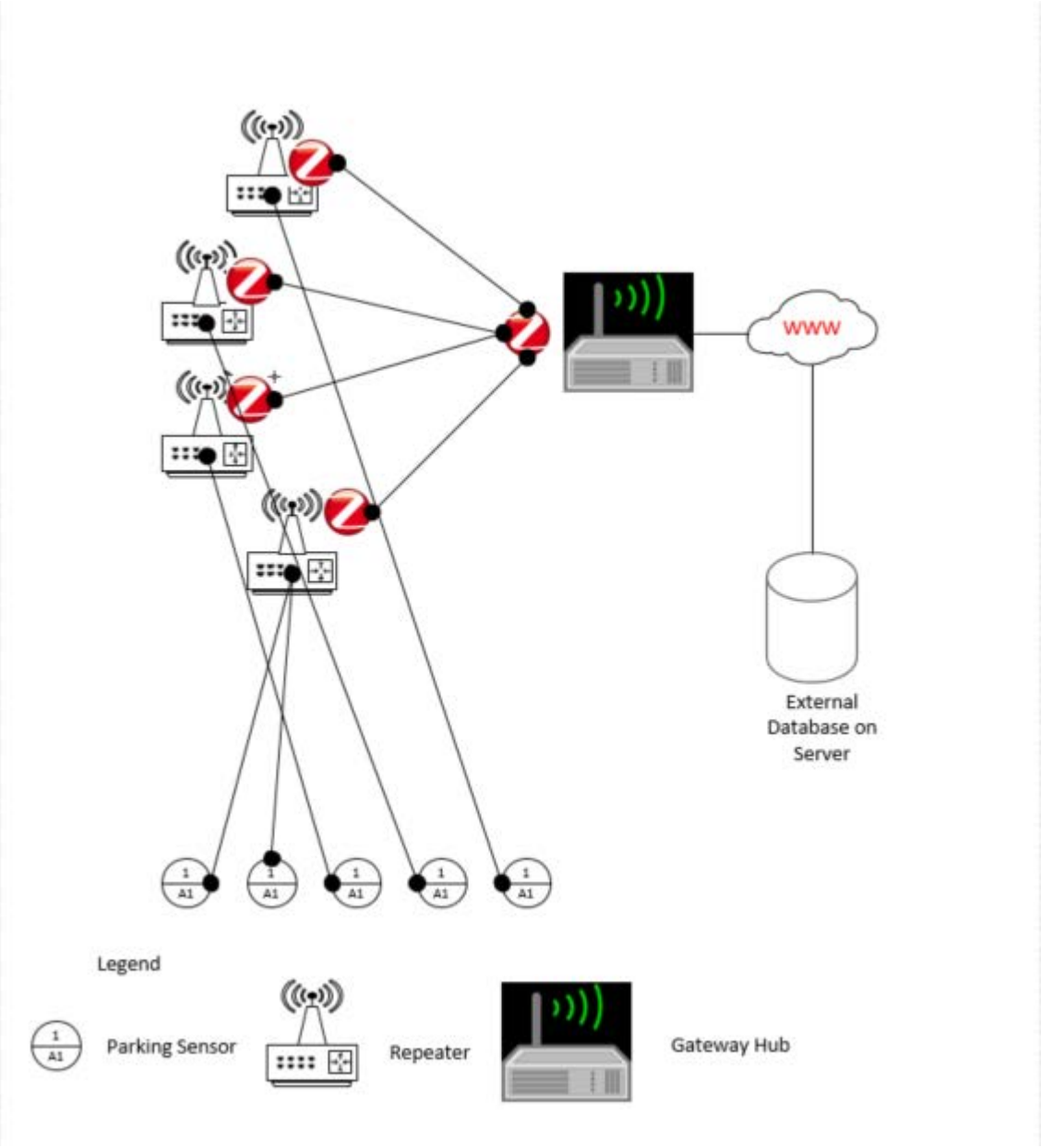

Figure 1-1 Parking sensors connecting to repeaters which then send the data to the gateway hub, then the gateway hub sends our data to our external server.

2. Product Description and Functional Overview

This section provides the reader with an overview of SmartPark the smart parking solution. The primary operational aspects of the product, from the perspective of end users, maintainers, and administrators are defined here. The key features and functionality found in the product, as well as critical user interactions and user interfaces are described in detail.

2.1 Features and Functions

The SmartPark system will consist of parking sensors, microcontrollers, and it will work in tandem with a web application and an Android application. The parking sensor node will be enclosed in a secure package either cylindrical or square to protect the electrical components which will be placed in the parking space. It will provide users the spaces in a parking lot which are opened and closed so users can reduce the frustration of looking for parking.

The parking sensors will generate data relevant to the specific parking space that it is located in. It will track things such as availability (if a car is there or not), time, among other administrative things. The data that is generated is a raw stream of data which will then be sent to a repeater if necessary. The repeater will then repeat the signal using a communication protocol until it reaches our Gateway hub so then it can send that data to our external web server and database for processing. The data that is cleaned up on the server will then be sent to the web and Android application to be parsed and presented to the user in a clean way. The two types of messaging protocols that we are discussing are JSON and XML.

The Android application will show the users on SmartPark enabled lots which spaces are open. The application will also allow the user to enter their schedule manually or if permission is granted we can read their local calendar from their phone. In addition, when the user nears the desired destination it will programmatically alert the user with a notification that there is parking near their desired location. The user will be able to turn this feature off if they opt out. The Android application will also allow you to do an address search to check if any lots surrounding your desired destination are SmartPark enabled to help in planning.

2.2 External Inputs and Outputs

SmartPark will have a parking sensor node which will receive an input to turn it on/off. Once it is on the node will receive its input from a vehicle which will serve as our vehicle detection mechanism. With the data generated from the vehicle and the sensor, the sensors will then communicate this signal to either a repeater or our gateway hub. The output generated from our hub and server will serve as the input for our Android and web application. The Android application will need to take in GPS sensor data so our application knows its location. In addition, a user will have to enter an address into the text field on the map to get information on the parking spaces if it’s a location that they are not currently in.

Table 2-1 External Inputs and Outputs

Name	Description	Use
Power Button	A button located on the board to power the node. It will turn the board on or off.	**Input:** It will be initiated by the user or whoever is installing the sensor when they are ready to deploy the node.
Vehicle	The vehicle will be detected by our sensor and detect its presence.	**Input:** It will be initialized when the driver parks in a space and the sensor will generate data on that event.
User input in web and Android Application	The user will have to enter an address so we can show them SmartPark enabled parking lots.	**Input:** Through phones keyboard user will enter and username, password information and an address.
Phone GPS Sensor Data	The GPS Sensor data will generate information on the location of the user.	**Input:** GPS Sensor data on their location.
Presentation of open and closed parking spaces	Will show green if the space is available.	**Output:** Will show the user availability of parking spaces.

2.3 Product Interfaces

2.3.1 Android Application

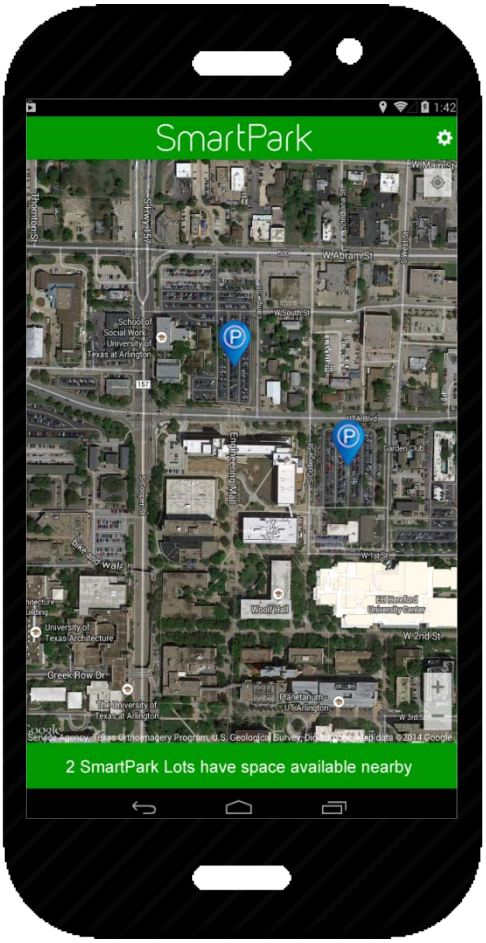

Figure 2-1: Overview of SmartPark enabled lots

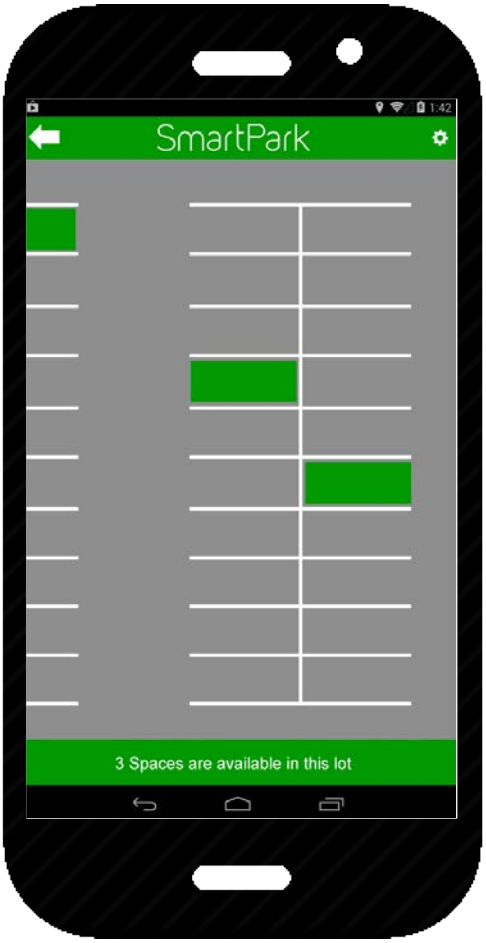

Figure 2-2: Detailed view of parking lot

3. Customer Requirements

This section consists of requirements provided by our sponsor Ravi Kant and Team Sense that must be implemented in order to provide a complete product. Each requirement will have a description, a source, constraints, and a priority. The modification of these requirements will require the approval of the team and sponsor.

3.1 Administrator Portal

3.1.1 Description: SmartPark will have an administrator portal for Valet managers to show the status of each of the parking nodes in the parking lot, for example, battery status, network status, and vacancy.

3.1.2 Source: Sponsor

3.1.3 Constraints: Interference in the environment or in the radios may cause our radio transmission range to decrease which may limit our ability to connect the network of nodes.

3.1.4 Standards: Wi-Fi, Bluetooth, ZigBee (Standard will depend on which one we choose for our communication protocol.)

3.1.5 Priority: 4 - Low

3.2 Display Location of Spots

3.2.1 Description: SmartPark will be able to show which parking spots are taken or available through a graphical user interface. The red color will show the taken spaces and the green color will show the available spaces.

3.2.2 Source: Sense

3.2.3 Constraints: Interference in the radio signal or environment may limit our ability to generate sensor data and transmit it to be able to show it to the user.

3.2.4 Standards: Wi-Fi, Bluetooth, ZigBee (Standard will depend on which one we choose for our communication protocol.)

3.2.5 Priority: 1 - Critical

3.3 Suggest Parking by User's Location

3.3.1 Description: SmartPark should give a recommendation to the user of the Top 5 parking spots that are available according to the location where the user needs to be. Shortest distance to the location would be the basis for the recommendation.

3.3.2 Source: Sponsor

3.3.3 Constraints: Environmental constraints such as construction might generate too much interference for our radio to transmit the signal.

3.3.4 Standards: None

3.3.5 Priority: Future

3.4 Search Parking Lot by Address

3.4.1 Description: SmartPark should allow the user to search for an address and show them SmartPark enabled parking lots around the address. This will help in planning for parking at a location.

3.4.2 Source: Sponsor

3.4.3 Constraints: If Google Maps API endpoints are not accessible or they do not allow the use of their API.

3.4.4 Standards: None

3.4.5 Priority: 4 - Low

3.5 Synchronize with User's Calendar

3.5.1 Description: SmartPark should synchronize the calendar that is on the user's phone through the Google Calendar API. If the user does not allow permission to the application to access their Google Calendar there should be an option to manually add a schedule.

3.5.2 Source: Sense

3.5.3 Constraints: Google not allowing us to connect to their Google Calendar API endpoints or they deprecate their API.

3.5.5 Priority: 3 - Moderate

3.6 Alert User of Open Space Based On Schedule/Location

3.6.1 Description: SmartPark should alert the user, based off the schedule provided, that parking that is available through a notification. When the notification is clicked it would display which space is open. If done by location, SmartPark should notify the user when they get near their location.

3.6.2 Source: Sponsor

3.6.3 Constraints: Accurate location data is needed to obtain an accurate distance. Environmental factors may hinder accurate location data.

3.6.4 Standards: None

3.6.5 Priority: 3 - moderate

3.7 Easy to Add New Lots

3.7.1 Description: SmartPark should be robust to make it easy to deploy new lots to the existing network.

3.7.2 Source: Sponsor

3.7.3 Constraints: Environmental factors might hinder our transmission range to connect more nodes to the existing wireless sensor network.

3.7.4 Standards: None

3.7.5 Priority: 4 – Low

3.8 Easy to Add New Spaces to Existing Lots

3.8.1 Description: As a valet parking manager, SmartPark should make it easy to add a new space to an existing parking lot. It should add it to my existing network.

3.8.2 Source: Sponsor

3.8.3 Constraints: Environmental factors will interfere with our transmission which may limit our ability to connect to the existing network.

3.8.4 Standards: None

3.8.5 Priority: 4 - Low

3.9 User-Friendly Interfaces

3.9.1 Description: SmartPark should provide a graphical user interface that is simple to use and easy to navigate.

3.9.2 Source: Sponsor

3.9.3 Constraints: Google Maps API may limit our ability to present this information to the user in an intuitive way. Their focus is showing us a high level overview of the area, while SmartPark wants more detail.

3.9.4 Standards: None

3.9.5 Priority: 1 - Critical

3.10 Durable Design on Vehicle Sensors

3.10.1 Description: SmartPark should provide parking nodes of durable design. This will minimize the maintenance costs for the customer who purchased the system.

3.10.2 Source: Sponsor

3.10.3 Constraints: Interference in our packaging with our signal may limit our ability to house the sensor in a durable packaging.

3.10.4 Standards: None

3.10.5 Priority: 3 - Moderate

3.11 Repeater Nodes

3.11.1 Description: Repeater nodes shall be placed in different locations around the parking lot, they will repeat the signal generated from the parking sensor nodes until it reaches the hub that is connected to the internet. These will be powered by either batteries or solar panels.

3.11.2 Source: Sense

3.11.3 Constraints: Placement of repeater nodes and range of signals is dependent on physical/environmental characteristics of the parking area.

3.11.4 Standards: ZigBee/WiFi/Bluetooth

3.11.5 Priority: 1 - Critical

3.12 Hub Functionality

3.12.1 Description: The “hub” shall receive the data that has been repeated by our repeater nodes, the hub will be connected to the internet and it will send this data to the database on our server. The hub will serve as the gateway between the internet and the sensor network.

3.12.2 Source: Sense

3.12.3 Constraints: Placement of the gateway hub is dependent on physical/environmental characteristics of the parking area.

3.12.4 Standards: Cable - Broadband Internet Connection or a Wireless Internet Connection

3.12.5 Priority: 1 - Critical

4. Packaging Requirements

The vehicle sensors will come pre-assembled and ready to install. Each sensor will be packaged individually. The system will come complete with the sensors and all networking and control systems that are required. Both administrators (those installing and managing the parking sensor system) and users (those using the companion application to find parking) must create an account before using the system.

4.1 Google Play Publication

4.1.1 Description: The accompanying Android application will be available as a free download via the Google Play store.

4.1.2 Source: Sense

4.1.3 Constraints: The application must follow Google Play criteria for publishing applications. Android devices using the app must be version 4.0 or higher.

4.1.4 Standards: Google Play publishing guidelines and policies.

4.1.5 Priority: 4 - Low

4.2 User Web Application

4.2.1 Description: The web application will provide the same services as the Android application, except it will be available through a Web browser.

4.2.2 Source: Sponsor

4.2.3 Constraints: Application can only be used with a web browser that can execute JavaScript properly.

4.2.4 Standards: None

4.2.5 Priority: 1 – Critical

4.3 Sensor Housing

4.3.1 Description: The components of the vehicle sensor will be enclosed in a housing to protect them from being damaged by impacts and vehicle pressure. The housing will be waterproof to prevent internal components from short circuiting due to water damage. Internal components may be accessed by removing the case lid.

4.3.2 Source: Sense

4.3.3 Constraints: The case must not interfere with wireless signal from the internal electronics.

4.3.4 Standard: None

4.3.5 Priority: 1- Critical

4.4 User Manual

4.4.1 Description: A user manual will be provided with the parking system in both printed and digital form. This manual is intended for administrators of the parking system and will give details on how to maintain the sensor network, as well as how to use the admin web portal. It will contain information about product features, installation steps, battery replacement, and general troubleshooting help.

4.4.2 Source: Sense

4.4.3 Constraints: The manual will be provided in English only.

4.4.4 Standards: None

4.4.5 Priority: 2 - High

5. Performance Requirements

This section provides requirements for operation and performance of SmartPark system. It covers performance requirements for both hardware and software components of the system. Performance requirements encompass quality, responsiveness, reliability, and accuracy of the system.

5.1 Operating Conditions and Robustness

5.1.1 Description: SmartPark hardware system shall operate accurately and reliably in all weather conditions and it shall withstand force or pressure exerted by the vehicles.

5.1.2 Source: Sense

5.1.3 Constraints: None

5.1.4 Standards: None

5.1.5 Priority: 1 - Critical

5.2 Battery Life

5.2.1 Description: The battery used to power the hardware shall last for at least 1year.

5.2.2 Source: Sense

5.2.3 Constraints: Battery life may be variable depending on the power consumption of the specific hardware components used.

5.2.4 Standards: None

5.2.5 Priority: 3 - Moderate

5.3 Low Battery Level Detection

5.3.1 Description: SmartPark hardware system shall detect low battery level.

5.3.2 Source: Sense

5.3.3 Constraints: None

5.3.4 Standards: None

5.3.5 Priority: 4 - Low

5.4 Low Battery Performance

5.4.1 Description: SmartPark hardware system shall be fully functional even after the low battery level is detected by the system.

5.4.2 Source: Sense

5.4.3 Constraints: Maintenance crew replaces battery before it totally dies out.

5.4.4 Standards: None

5.4.5 Priority: 2 - High

5.5 Accurate Detection

5.5.1 Description: The hardware component of SmartPark system shall detect recently parked or departed vehicles accurately within minimal error rate. Error rate in this case is the measure of how often the system displays incorrect parking information to the user.

5.5.2 Source: Sense

5.5.3 Constraints: Failing or defective sensors may limit accuracy.

5.5.4 Standards: None

5.5.5 Priority: 2 - High

5.6 Real-Time Response

5.6.1 Description: SmartPark system shall send the information of the recently taken or vacated parking spot to the wireless network layer after the system confirms that the vehicle is parked or not parked.

5.6.2 Source: Sense

5.6.3 Constraints: None

5.6.4 Standards: None

5.6.5 Priority: 2 – High

5.7 Gateway Hub Performance

5.7.1 Description: The wireless network layer shall relay parking data to the centralized server immediately after receiving data.

5.7.2 Source: Sense

5.7.3 Constraints: A high amount of network traffic and congestion may limit our ability to immediately relay our data to our centralized server.

5.7.4 Standards: None

5.7.5 Priority: 2 - High

5.8 Android Application Map Display Response Time

5.8.1 Description: It shall take no longer than 10 seconds to update the display of available or taken parking spots on the map on users' mobile phones.

5.8.2 Source: Sense

5.8.3 Constraints: A congestion of web traffic may limit our ability to display the map in the amount of time stated. Also, a decrease of performance of the server might limit this ability as well.

5.8.4 Standards: None

5.8.5 Priority: 3 - Moderate

5.9 Web Authentication Duration

5.9.1 Description: The web management application will attempt to authenticate the administrative user for 20 seconds. If the connection could not be established within 20 seconds, the web user interface would halt login activity and display connection time out error message.

5.9.2 Source: Sense

5.9.3 Constraints: A congestion of web traffic may limit our ability to display the map in the amount of time stated. Also, a decrease of performance of the server might limit this ability as well.

5.9.4 Standards: None

5.9.5 Priority: 3 - Moderate

6. Safety Requirements

This section covers the safety requirements the system must fulfill. These requirements will cover the safety of both the user and the system. The system must be safe in regards to hardware, software and the user's overall safety.

6.1 Heat Dissipation

6.1.1 Description: The system shall dissipate heat produced by the components of the system to prevent overheating.

6.1.2 Source: Sense

6.1.3 Constraints: None

6.1.4 Standards: None

6.1.5 Priority: 2 – High

6.2 Application Disclaimer

6.2.1 Description: The system shall have a disclaimer that warns the user of not using the product while driving to avoid accidents. This disclaimer will only be in English.

6.2.2 Source: Sense

6.2.3 Constraints: None

6.2.4 Standards: None

6.2.5 Priority: 1- Critical

6.3 Static Electricity

6.3.1 Description: The system shall be able to handle static electricity produced from its surrounding environment and the packaging components should not produce static electricity.

6.3.2 Source: Sense

6.3.3 Constraints: None

6.3.4 Standards: None

6.3.5 Priority: 3-Moderate

7. Maintenance and Support Requirements

This section describes the requirements for maintenance and support of the product after delivery. It covers specific requirements on how the system can be maintained, troubleshot, and scaled in the future. It provides maintenance and support information of both hardware and software subsystem.

7.1 Battery Replacement

7.1.1 Description: The battery on SmartPark hardware system shall be easily replaceable without the need of disassembling the product.

7.1.2 Source: Sponsor

7.1.3 Constraints: Battery must be of correct size and type

7.1.4 Standards: None

7.1.5 Priority: 2 – High

7.2 Scalability

7.2.1 Description: SmartPark system shall provide support for easy scalability to expand the coverage area in the future. In other words, it shall be easy to add or delete a new hardware node in the parking lot.

7.2.2 Source: Sponsor

7.2.3 Constraints: None

7.2.4 Standards: None

7.2.5 Priority: 4 – Low

7.3 Software Maintenance

7.3.1 Description: All source code files written by the team and related documentation prepared by the team will be provided for future software maintenance.

7.3.2 Source: Sense

7.3.3 Constraints: None

7.3.4 Standards: None

7.3.5 Priority: 3 – Moderate

7.4 Sensor Maintenance

7.4.1 Description: The vehicle detection controller shall be reprogrammable in case we need to deploy bug fixes or if upgrades become necessary. The vehicle detection sensors shall be replaceable in case of hardware failure.

7.4.2 Source: Sense

7.4.3 Constraints: None

7.4.4 Standards: None

7.4.5 Priority: 4 - Low

8. Other Requirements

This section covers any requirements necessary to project that are not covered in the previous sections.

8.1 Ease of Installation

8.1.1 Description: The system components shall be easy to install, i.e. it should be non-intrusive, and disruption of business should be minimal.

8.1.2 Source: Sense

8.1.3 Constraints: None

8.1.4 Standards: None

8.1.5 Priority: 3 - Moderate

8.2 Unique Sensor ID

8.2.1 Description: The system shall have sensors with unique IDs that help in identifying the parking lot space. The Unique Sensor ID will be sent to the server for later processing.

8.2.2 Source: Sense

8.2.3 Constraints: None

8.2.4 Standards: None

8.2.5 Priority: 1 - Critical

8.3 Log Files

8.3.1 Description: The system shall generate log files that will be referenced for debugging.

8.3.2 Source: Sense

8.3.3 Constraints: None

8.3.4 Standards: None

8.3.5 Priority: 3 – Moderate

8.4 On/Off Button

8.4.1 Description: The vehicle detection controller should have a power button to turn it on or off.

8.4.2 Source: Sense

8.4.3 Constraints: None

8.4.4 Standards: None

8.4.5 Priority: 3 – Moderate

8.5 Wireless Sensor Network

8.5.1 Description: The SmartPark system shall implement a wireless sensor network for the communication between all the parking nodes. It will implement either a multi-hop network topology or a direct access network to our gateway hub.

8.5.2 Source: Sense

8.5.3 Constraints: Interference in the environment might limit our ability to connect nodes to the network.

8.5.4 Standards: ZigBee, Wifi, Bluetooth

8.5.5 Priority: 1- Critical

8.6 Gateway Hub Status Indicators

8.6.1 Description: The SmartPark system gateway hub should have light indicators that display healthy connectivity to the internet.

8.6.2 Source: Sense

8.6.3 Constraints: Interference in the environment may limit our ability to connect nodes to the network.

8.6.4 Standards: Wi-Fi, Direct Connection to Internet

8.6.5 Priority: 4 - Low

9. Acceptance Criteria

The criteria below serve as a means of demonstrating the functionality of the product to the sponsor and customer. Members of the team will be used to test the system and ensure that these requirements are met.

9.1 Verify that the Android app determines a user's current location

9.1.1 Requirement(s) addressed: Implied in various requirements.

9.1.2 Verification Procedure: Members of the team will search for their location from different areas to ensure an accurate display on the map.

9.2 Verify that the app shows open parking spaces

9.2.1 Requirement(s) addressed:

3.2	Display Location of Spots
3.3	Suggest Parking by User's Location

9.2.2 Verification Procedure: This requirement will be verified by zooming in on the test lot and verifying that the current status (occupied/vacant) of the test lot is accurately reflected in the in both the Web and Android applications.

9.3 Verify that the app shows information about specific lots at the appropriate time.

9.3.1 Requirement(s) addressed:

3.2	Display Location of Spots
3.3	Suggest Parking by User's Location
3.5	Synchronize with User's Calendar
3.6	Alert User of Open Space Based on Schedule/Location

9.3.2 Verification Procedure: This will be verified by entering destination and time information into the system as part of a schedule and verifying that a notification is displayed at the appropriate time and contains information about relevant parking lots.

9.4 Verify that the Admin console displays accurate information about sensor and network status.

9.4.1 Requirement(s) addressed:

3.1	Administrator Portal

9.4.2 Verification Procedure: This will be verified by accessing the Admin console. Sensor status will be checked by replacing the battery with a weak one, and ensuring a low battery warning is displayed in the console. Network status will be verified by blocking the signal to one of the sensors and ensuring a warning status is displayed in the console.

9.5 Verify that the system is user friendly.

9.5.1 Requirement(s) addressed:

3.7	Easy to Add New Lots
3.8	Easy to Add New Spaces to Existing Lots
3.9	User-Friendly Interfaces

9.5.2 Verification Procedure: The Website Application and Android application will be thoroughly tested to ensure a fluid flow and ease of use. Members of the target demographic will be shown the system to guarantee they are intuitive and easy to interact with. The Admin console will be shown to the sponsor to verify ease of use.

9.6 Verify that the system correctly authenticates account login.

9.6.1 Requirement(s) addressed:

5.9	Web Authentication Duration

9.6.2 Verification Procedure: This will be tested as both a user and an administrator. Valid and invalid credentials will be tested to ensure the system handles each case appropriately.

9.7 Verify that accounts can be created successfully.

9.7.1 Requirement(s) addressed:

5.9	Web Authentication Duration

9.7.2 Verification Procedure: Both user and administrator accounts will be tested. Different information will be entered during the account creation process to ensure that correct information is accepted, and incorrect information is rejected. Accounts that have been registered will then be checked to ensure they have been successfully created and stored in the system.

10. Use Cases

This section describes how users and administrators interact with the system. These cases specify different actions that may be taken by users and/or administrators of the system. This section begins with a UML diagram outlining the overall system use. Each use case consists of a scenario for the case, the actors involved, what the use case begins with (TUCBW), and what the use case ends with (TUCEW).

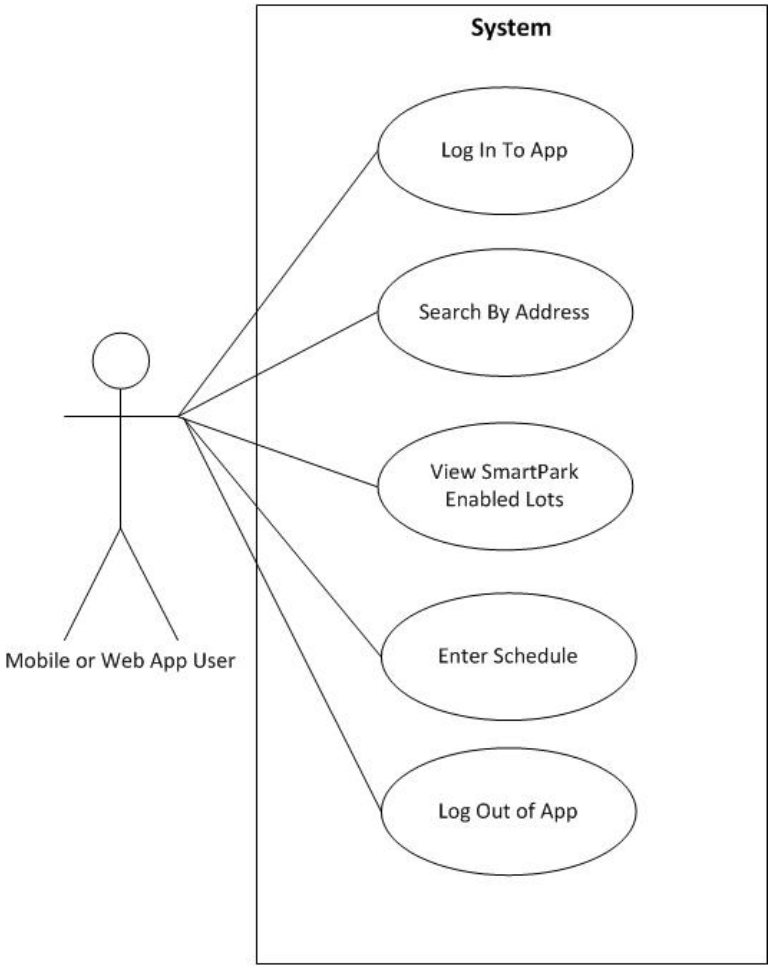

Figure 10-1 Use Case Diagram for Mobile or Web App User

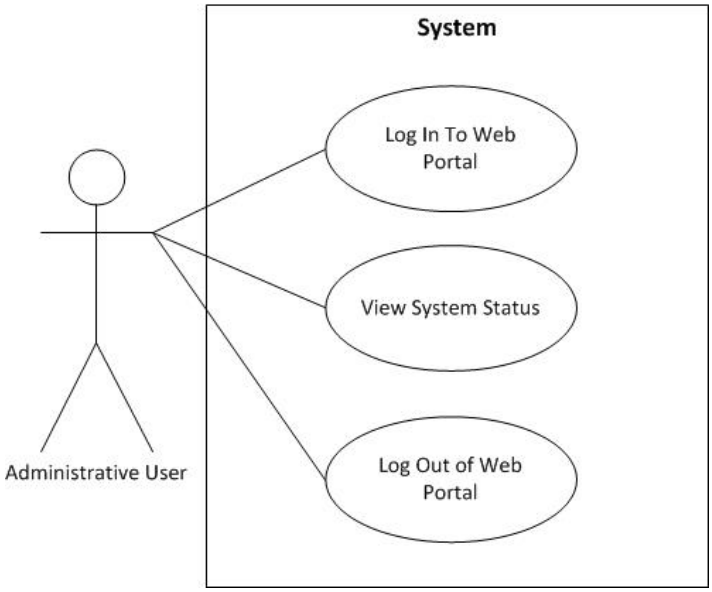

Figure 10-2 Use Case Diagram for Administrator

10.1 Login to App

10.1.1 Scenario: The user selects the application icon on their device, followed by them entering their username and password and clicking the login button. The app would then place a reference point for the user's current location on the map.

10.1.2 Actor(s): User, Authentication Service

10.1.3 TUCBW: The user selects the application icon on their device.

10.1.4 TUCEW: The system displays the user's current location.

10.2 Search by Address

10.2.1 Scenario: The user activates the search bar, types and address, then submits the address. The app would then respond by displaying all SmartPark enabled lots that are in close proximity to that address.

10.2.2 Actor(s): Users, Location Service

10.2.3 TUCBW: The user click on the address bar

10.2.4 TUCEW: The system displays SmartPark lots near the address entered.

10.3 View SmartPark enabled Lots

10.3.1 Scenario: The User is presented with a map showing all SmartPark enabled lots near their current location or address. The user selects one of the lots, and the app view changes to display a detailed view of the lot, highlighting empty spaces.

10.3.2 Actor(s): User

10.3.3 TUCBW: The user selects a SmartPark lot on the map.

10.3.4 TUCEW: The system displays a detailed view of the lot.

10.4 Enter Schedule in Application

10.4.1 Scenario: A user is logged into the mobile application. They touch the applications settings button. There will be a section for schedule in the settings. The schedule fields will include event title, time, location, and days of the week they want this event to happen. The user can touch “+” to add another event or finish to save the schedule.

10.4.2 Actor(s): User

10.4.3 TUCBW: The user clicks on schedule.

10.4.4 TUCEW: The schedule event is added to the system.

10.5 Logout of App

10.5.1 Scenario: The mobile phone user is currently logged in to the SmartPark App and taps the logout button.

10.5.2 Actor(s): Mobile Phone App User

10.5.3 TUCBW: The user is currently logged in to the SmartPark App.

10.5.4 TUCEW: The user is logged out of the system.

10.6 Login to Web Portal

10.6.1 Scenario: The administrator enters the web address and logs in with the required credentials, and then an overview of the systems status is displayed.

10.6.2 Actor(s): Administrator

10.6.3 TUCBW: The administrator enters the web address and logs in.

10.6.4 TUCEW: The systems status is displayed.

10.7 Logout of Web Portal

10.7.1 Scenario: The administrator is currently logged in to the administrative web portal and clicks the logout button. The administrator is then logged out of the system.

10.7.2 Actor(s): Administrator

10.7.3 TUCBW: The administrative user is currently logged in to the SmartPark App.

10.7.4 TUCEW: The administrative user is logged out of the system.

11. Feasibility Assessment

This section will analyze the feasibility of the university touring app. We will cover the scope, research, technical analysis, cost, team resources, and estimation for the project. This assessment is based on the team's current knowledge and prior experience and is subject to change.

11.1 Scope Analysis

After analyzing the scope and having multiple discussions with the team and sponsor about the complexity of SmartPark we came to the conclusion that a commercial system will not be feasible for the projects allotted time. However, all of the critical requirements stated below are reasonable and the prototyping of these by the deadline seems feasible. Sense has estimated that the most work of our system will be in three of the critical requirements, which is the ability to detect a vehicle, send this data to the internet, and display it to the user all while keeping the sensor housed.

Even though a lot of our requirements have a critical priority, the development of some of these will not cause our system to fail. Based on these factors and to be able to show the completion of these requirements we are going to develop SmartPark using an incremental approach in order from critical to low priority. Our goal as a team will be to complete at least 100% of Phase 3 and if time and budget allows, start and finish Phase 4.

Priority - Critical (Core Functionality) – Phase 1

- Display location of spots
- User-Friendly Interfaces
- Repeater Nodes
- Hub Functionality
- User Web Application
- Sensor housing
- Operating Conditions and Robustness
- Application Disclaimer
- Wireless Sensor Network
- Unique Sensor ID

Priority – High (Performance and Packaging) – Phase 2

- User Manual
- Accurate Detection
- Real-time response
- Gateway hub performance
- Battery Replacement

Priority – Moderate (User Experience) – Phase 3

- Sync with users calendar through Google Calendar API
- Alert user of open space based on schedule/location
- Durable design on parking sensors

Priority – Low (Administrative Features) – Phase 4

- Administrative Portal
- Search Parking Lot By Address
- Easy to add new lots
- Easy to add new spaces to existing lots
- Google play publication
- Low Battery Level Detection
- Hardware System Setup Delay
- Scalability
- Sensor Maintenance
- Gateway Hub Status Indicators

Priority – Future (Recommendation Engine) – Phase 5

- Suggest parking by user's location

11.2 Research

Team Sense is still performing research on the components that we will use. So far, our team has done research focusing on core components of our project: hardware and network layer. We have found a Hall Effect sensor for the X-Y-Z axis which will help us in determining if a car is present. We have also researched network managers that will help us repeat the generated data to the gateway hub. As a team there is still extensive research to be done on wireless sensor networks and hardware components but the pieces are starting to come together as we spend more time on it. We also have found that each one of these parking sensor components will need a sensor, microcontroller, and some type of transmitting device which could be an antenna. We also must do more research on communication protocols such as ZigBee since that is the top choice for wireless sensor networks. Further research will be done in the future regarding the specifics of the system components as we get a clearer picture of the system components.

We have found several hardware components that might make our development time shorter and similar hobbyist projects that detect vehicles. One of the boards that we found is the Waspmote (Fig 11-1) which will enable us to integrate our sensors more easily. This board seems the most promising but the team is still exploring several different options.

Based on the research that we have conducted until now, we are gaining confidence in the feasibility of the project.

Figure 11-1 Waspmote Sensor board.

11.3 Technical Analysis

The technical aspects of our system will include, embedded systems, vehicle detection algorithms, and wireless sensor networks. We will also deal with cloud services to meet certain of our requirements. We recognize that as a team our hardware skills are not very strong, which is why we have identified that detecting a vehicle's presence and networking all of the parking spaces will be our biggest challenge. SmartPark will have to convert some type of analog signal to a digital one in a quick manner to give the real time response the system needs.

To help in overcoming these challenges, our team has spent extensive time researching different technologies and product lines to meet our requirements. Also, our sponsor Ravi Kant has experience in hardware systems, so we have resorted to him when we need guidance in some type of hardware technical question. We also hope to have the support of either our current Senior Design Teams or the staff at UTA if we were to need their guidance.

To ensure that this project is successful, the skills that we are going to need to obtain will be how to create and communicate with a wireless sensor network, also how to interact with the sensors on each of our logic boards, designing the hardware board itself, and writing good software to interface all of these things together. One of the core functionalities will be to receive the sensor data and deliver this information to the user so the user gets a real idea of where parking is located.

From our research we have discovered that there are similar systems that have set out to solve the parking problem, and we may be able to use some of their components and design to complete SmartPark and its requirements. Several companies also sell some of these components packaged and pre-built which should help us in reducing our development time. From a programming standpoint, we have discovered that we will have to use Python, C, or C++ for the programming of the hardware and some server side language such as Python, PHP, and C #for the processing of our data. Our database that will be sitting in the cloud will be some type of relational database; one of the options we are considering is MySQL.

11.4 Cost Analysis

The team has conducted the preliminary research of the possible hardware parts and other software components that SmartPark might use. We have based our research on the cost of components, which with our knowledge SmartPark will need to function. We have based these estimates on approximate ranges rather than absolute values because we have not finalized which parts we are going to use in SmarkPark. Based on the preliminary research, we have discovered that SmartPark will consist of several hardware peripheral devices; therefore, the budget allocated for our projects will mostly be spent on acquiring these devices. We have also found out that we have two choices on acquiring the hardware devices; we can either get the development kit board along with preinstalled sensors, which will have a higher cost, or we can get the separate microcontroller, sensors, and the development board, which will have lesser cost. In addition, the cost of each of these boards will restrict us from building out multiple parking spaces for SmartPark. In any case, the cost for acquiring the components will stay within our $800 budget to build the minimal required hardware system. In addition, we are also trying to get the development boards on discounted price or free of cost for educational purposes and we are currently negotiating with some of the companies, like Linear Technologies and Libelium Distributed Communications. In addition, we are researching more wireless sensor network vendors to broaden our options. The tables below provide the cost breakdown of several components that we have considered using in SmartPark:

Table 11-1: Development kit with preinstalled sensors

Components	Quantity	Unit Price ($)	Cost ($)
Wireless Sensor Network Starter Kit	1	800	800
Networking Hardware	2	50 -100	100 - 200
Web Hosting Service	1	5/month (60 /year)	60
Google Play Account	1	25	25
Total			**985 - 1085**

Table 11-2: Development kit without preinstalled sensors

Components	Quantity	Unit Price ($)	Cost ($)
Development Board such as Arduino, Raspberry Pi, BeagleBoard.	4	35-70	140 - 280
Vehicle detection sensor such as Hall Effect Sensor	4	5-10	20 - 40
Temperature Sensor	4	1-4	4 - 16
Networking Hardware	2	50-100	100 - 200
Battery	4	15-30	60 - 120
Web Hosting Service	1	5/month (60 /year)	60
Google Play Account	1	25	25
Antenna	4	50	200
Cables	1	10	10
Total			**619 - 951**

Based on the above price breakdown, our $800 budget will limit us from acquiring the development kit with preinstalled sensors; however, if we were able to get some discount from the manufacturer, we will be able to obtain it. On the other hand, it seems completely feasible to buy the development kit (without preinstalled sensors) and other components to build four units of parking space locating hardware.

11.5 Resource Analysis

Our team consists of one software engineer, two computer scientists, and one computer engineer. The computer engineer will be responsible for hardware design and implementation of the system. One of the computer scientists and the software engineer will be responsible for software design and implementation. The other computer scientist will be responsible for integrating hardware and software components as well as assisting the computer engineering in hardware implementation. Overall, all the team members will always be ready to help each other when needed.

The strength of the team is comprised of prior experience in software development. All of our team members have good knowledge of common programming languages like Java and C. One of the computer scientists and the computer engineering has some knowledge of Android application programming, so that is a big plus. One of the computer scientists has some knowledge of hardware design and implementation.

The primary weakness of the team is our relative inexperience of hardware devices, like microcontroller and sensors, which will be an integral part of the system. Besides that, our team only has a limited knowledge of quality assurance (test plans, test supervision and execution). However, all of our team members are competent enough to learn the required skills and technologies to complete the project successfully.

11.6 Schedule Analysis

The development team used two methods to estimate the schedule of the project. The first method that we used was Jones' First Order Estimation. In this method, we first calculated the function points of the project based on the following project's characteristics and the corresponding complexities that our team had identified as shown in the Table 11-3 below:

Inputs – Messages used to manipulate program's data.

Outputs – Messages that the program generates.

Inquiries – Input/output combinations in which an input generates simple output.

Logical internal files – Major logical groups of end-user's data or control information that are controlled by the program.

External interface files – Files controlled by other programs that this program uses; data or control information that enters or leaves the program.

Table 11-3: Function Points

	Function Points			
Project Characteristics	**Low Complexity**	**Medium Complexity**	**High Complexity**	**Total**
Number of inputs	2 * 3	2* 4	0 * 6	14
Number of outputs	3* 4	1 * 5	0 * 7	17
Inquires	1 * 3	2 * 4	1 * 6	17
Logical internal files	3 * 7	1 * 10	0 * 15	31
External interface files	2 * 5	0 * 7	0* 10	10
Total				**89**

The total unadjusted function points calculated from Table 11-1 is 89. The unadjusted function points total is heavily influenced by the number of logical internal files and their corresponding complexities.

Now, we will calculate Influence Multiplier to come up with our adjusted function point total. We used the following formula and the system characteristics as shown in Table 11-3 to come up with our adjustment factor (influence multiplier):

Adjustment Factor = (Total * 0.01) + 0.65

Table 11-4: Adjustment Factors

General System Characteristics	Effort (0 – 5)
Data Communication	4
Distributed data processing	0
Performance	4
Heavily used configuration	5
Transaction rate	4
Online data entry	1
End-user efficiency	4
Online update	1
Complex processing	2
Reusability	3
Installation ease	5
Operational ease	5
Multiple sites	1
Facilitate change	3
Total	42
Adjustment Factor	1.07

Adjustment Factor = (Total * 0.01) + 0.65 = (42 * .01) + 0.65 = 1.07

Now, we will calculate our adjusted function point total based on the following formula, and the corresponding result is shown in Table 11-3:

Adjusted Function Point Total = Unadjusted Function Point Total * Influence Multiplier

Table 11-5: Adjusted Function Point Total

Unadjusted Function Point Total	89
Influential Multiplier	1.07
Adjusted Function Point Total	95.23

Using adjusted function point total, we now can calculate the estimated duration of time that our project will take to complete. Table 11-4 shows the duration calculation using Jones' First Order Estimation for the Systems project category.

Table 11-6: Jones' First Order Estimation

	Best in class	**Average**	**Worst in Class**
Adjusted Function Points	$95.23^{0.43}$	$95.23^{0.45}$	$95.23^{0.48}$
Total	7.09 months	7.77 months	8.9 months

Analyzing the above estimates, we conclude that it will approximately take 7 calendar months to finish our project in the best case, whereas it will take about 9 months to finish the project in the worst case scenario. However, we also realize that we are estimating based on our limited knowledge at this early stage of our product development phase. And we are very optimistic that the estimates will get better and will be refined along various phases of our project.
The second method that we used to estimate the schedule of our project is Basic CoCoMo method. The team has realized that our project best suits semi-detached project type, so the following highlighted coefficients were used:

Table 11-7: Basic COCOMO Estimation Coefficients

Coefficient	a	b	c	d
Organic	2.4	1.05	2.5	0.38
Semi-detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Also, the following formulas were used to calculate the estimated effort and duration for our project:

i. Effort (in Person Months), $E = a(KSLOC)^b$
ii. Duration (in Months) = $c(Effort)^d$

The team has evaluated that there will be approximately 3000 source lines of codes (SLOC) for aggressive estimate and 4000 SLOC for conservative estimate.

For aggressive estimate:
Effort = $3(3)^{1.12} = 10.27$
Duration = $2.5(10.27)^{0.35} = 5.64$ months
For conservative estimate:
Effort = $3(4)^{1.12} = 14.17$
Duration = $2.5(14.17)^{0.35} = 6.32$ months

The summary of calculations is shown in Table 11-5.

Table 11-8: COCOMO Estimation (semi-detached)

Basic COCOMO Estimation	Aggressive	Conservative
Effort (Person Months)	10.27	14.17
Duration (Months)	5.64	6.32
People Required	1.82	2.24

The effort calculation using COCOMO method assumes full-time person months; however, all of our team members will be taking full-time course loads, so we will not be able to work 40 hours per week on our project (We estimate that we will be able to work between 10 - 20 hours per week for this project). The table above shows that the project requires approximately 2 people to complete the project in calculated duration. Nevertheless, there are four members in our team and our combined work effort should compensate the full-time work assumed by the formula. Therefore, the formula mentioned above should approximately calculate our project duration. From the above estimates, we conclude that our project will take close to five and a half months to completion on an aggressive end, whereas it will approximately take six and a half months to completion on the conservative end.

In order to get an average estimate from the above two methods, the development team has conducted one more estimation calculation using Sanity Test (Weiss &Wysocki). The formula used to calculate schedule estimates for this method is as follows:

$$E = \frac{O + 4M + P}{6}$$

In the formula, E is estimated duration, O is optimistic duration, M is nominal duration, and P is pessimistic duration.

From above estimates, our optimistic duration is 5.64 months, nominal duration is 7.06 months and pessimistic duration is 8.9 months. Therefore, our averaged estimate will be:

$$E = \frac{5.64 + 4 * 7.06 + 8.9}{6} = 7.13\ months$$

Based on the above Sanity Test result, the project is estimated to be completed within 7.13 months. However, we will only have approximately 6 months period for our Senior Design I and Senior Design II courses. Nevertheless, we strongly believe that we may have overestimated number and the complexities of the project characteristics of function points, and we also believe that we will be able to add some extra work hours between the semester breaks and in the weekends during the semester. Also, the estimates will get refined to a more feasible number as we get more familiar with our product. In addition, we will be classifying our requirements based on the priority and implementing the critical requirements first. So we still believe that we will be able to complete the critical and most of the high priority requirements before Senior Design II semester ends.

Considering all of the feasibility analysis and also considering that we will be following the designed-to-schedule software lifecycle approach for our implementation phase, we believe that our project is very much feasible, or at least we will be able to finish most of the critical and high priority requirements within the allotted budget and timeframe.

12. Future Items

This section focuses on features we would have liked to implemented, but due to time and budgetary constraints and other factors, we decided it will be in the project's best interest not to implement these features. These features may be implemented in future versions of the product.

12.1 Suggest Parking by User's Location

12.1.1 Requirement Description: As a university student, SmartPark should give me a recommendation of the Top 5 parking spots that are available according to the location where I need to be. Shortest distance to the location would be the basis for the recommendation

12.1.2 Constraint: We will not have enough nodes deployed which will result in not enough data generated to make smart recommendations.