Park Shark

University of Central Florida

Department of Electrical Engineering and Computer Science

Senior Design 2, Fall 2018

Group D

Travis Bangs	Computer Engineering
Keegan Conway	Computer Engineering
Marcelino Galarza	Computer Engineering
Beatriz Jimenez	Electrical Engineering

Table of Contents

List of Figures

List of Tables

1.0 Executive Summary

One of the hardest challenges every student will encounter throughout their academic career at the University of Central Florida is finding a parking spot on-campus. UCF currently has a system in place that attempts to monitor the availability of parking. However, this system is inaccurate and often leaves students scrambling to find a spot before their class starts. The idea behind Park Shark is to create a system that will report an accurate number of available parking spots left and be able to communicate this information to students quickly and effectively via the internet.

The goal of this project is to use two methods for detecting spots simultaneously: computer vision and ultrasonic sensors. Using computer vision detection allows keeping watch over multiple spots, reporting availability. Being able to monitor multiple spots will reduce the cost of our system compared to other systems that require monitoring sensors for each individual parking spot. If only computer vision is used to detect availability, issues arise with spots that may not be visible from the camera's position. The solution is to use wireless monitoring sensors placed in specific problematic spots. This method allows accurate reports of availability for spots that are unable to be monitored by a camera, such as corners of garages as well as the roof of the parking garage.

This system will be designed as a proof of concept. The final demo will utilize one computer vision module and several sensor modules outfitted at a parking garage on UCF's campus. This product is designed with market value in mind. If we can create a system that monitors multiple parking spots efficiently for far less cost than current systems being used by companies such as Disney, then our project could be marketed to companies in a need of a cheap and efficient parking monitor system.

The first deliverable of this project will be a small, durable device that will house the computer vision system. This system will include a Raspberry Pi, a camera in conjunction with OpenCV. This system should be able to communicate to report if a parking spot is available. This system will also act as the "master node", sending wireless sensor data to the server. The second deliverable of this project will be the wireless sensor module. This system will include a Nordic nRF52832 Bluetooth transceiver, an ultrasonic sensor, and a small power supply with batteries. This system will communicate wirelessly to report if a parking spot is available or unavailable. The power requirement for this system will be much less than the camera system, as running the ultrasonic sensor requires much less power.

Our final deliverable will be a mobile app and website. These applications will collect and display the parking availability information. The system will be able to display the number of spots available on a floor and analyze peak times to give students an estimated amount of time to find a parking spot. The website and app will be able to be used on any modern computing device and smartphone.

2.0 Project Description

The Park Shark system will be a new way for students to receive up to date parking availability information. This system will replace an old system that reported inaccurate information and was of no benefit to the student populace.

This section will discuss:

- The motivation and influences behind the project and the members responsible for the various deliverables
- The goals and objectives of our project
- The requirements and specifications for the entire system
- The House of Quality

2.1 Project Motivation

The main motivation for this project is an improvement of an already existing system. UCF has a monitoring system in place, as well as an application and website to display the information. However, this system is not accurate and is hardly used by the student population. UCF's system monitors the parking availability by using sensors at the entrance to each garage, attempting to monitor the availability by tracking the total number of cars currently present. However, this method introduces a large amount of reliability and accuracy problems. Our system will attempt to rectify this issue by using a multi-sensor approach. The Park Shark parking system is heavily inspired by others currently available on the market today, such as the system used in the parking garages at Disney Springs, which utilizes a computer vision approach to detecting vehicles present in parking spots. The group will mitigate the flaws of this system by introducing a corollary, lower cost method of detection via wireless sensors place-able at spots that cannot be covered by the field of view of the camera system. These sensors as well as the camera's ability to monitor multiple spots from one unit reduces the required amount of hardware and allows clients to monitor their parking with a far lower upfront cost, and still be able to scale the system to their own needs and budget concerns.

2.2 Goals/Objectives

The final product produced should be one that is able to report parking availability in an accurate and reliable way, as well as be as cost effective as possible. The product should be able to report this data via the internet in a simple and aesthetically pleasing way, ensuring the student population is able to access the information as fast as possible. The hardware aspects should be durable and reliable, ensuring that both the computer vision and hall effect system operate without need for large amounts of setup or calibrating once they are installed. The website and mobile application should be able to display the data in near real time and be as simple and easy to use as possible. The ease of use should apply both to the end user and to the administrators who configure the system and change settings.

2.3 Requirements and Specifications

This section will detail the requirements that must be satisfied for our project. The following tables will be used in the design and referenced during testing to ensure we meet requirements set at the beginning of our project.

Table 1: Requirements for computer vision system.

Small form factor	i.	Dimensions of product should not exceed 2.9 x 8.9 x 6.5 inches
	ii.	Weight of product should not exceed 10 lbs
Low cost	i.	The production of a single vision system should not exceed $200
	ii.	No single component should exceed $150
Power	i.	This module shall pull no more than 300 mA
	ii.	This device will be run off a small battery pack, and will require a recharge/change every 24 hours
Communication	i.	Be able to receive sensor reading wirelessly via Bluetooth
	ii.	Transmit data to database over WiFi
Software	i.	The system should be able to export the availability of individual spots to the database
	ii.	The method used for detecting vehicles should be accurate to a percentage of greater than 90%
	iii.	The sensor should be able to provide availability for a minimum of 3 spots

Table 2: Requirements for wireless sensor system.

Small form factor	i.	Dimensions of product should not exceed 4 x 3.7 x 3.1 inches
	ii.	Weight of product should not exceed 3 lbs
Low cost	i.	The production of a single system should not exceed $50
Operating distance	i.	Unit should be able to detect vehicles at least 5 ft. from the wall or the ceiling
Communication	i.	Unit should be able to transmit data wirelessly via Bluetooth
Power	i.	This module shall pull no more than 200 mA
	ii.	This device will run off a small battery pack
	iii.	The device should enter a power saving mode in between taking sensor readings

Table 3: Requirements for website and mobile application.

User Interface	i.	Display information in a simple, effective way
	ii.	Allow users to access parking availability
	iii.	Allow admins to login to administrator account
	iv.	Allow admins to add, delete, close, and open parking garages
Parking info	i.	Refresh all parking info every 30 seconds
	ii.	Allow functionality to show parking info on any amount of parking garages
	iii.	Display parking statistics such as peak hours and average parking time
Technology	i.	Website and Mobile App should run on any modern device
	ii.	The website should be able to be viewed in any desktop or mobile browser
Database	i.	A database that stores all parking data as well as login credentials should be created
	ii.	The website and app should use the same database to access data
	iii.	The database should be able to be accessed from the Computer Vision system to be able to send parking data to.

The above tables represent design requirements that were created during initial system design. Changes to requirements and design may occur during the prototyping and design portion of this project. Any changes to these requirements should be discussed and updated accordingly. If the need to change a design requirement does occur, the process for updating requirements is as follows:

- Discuss the design issue and discuss possible solutions to attempt to rectify and keep design requirements
- If the issue is determined to be an overwhelming problem for the entire system, then an update to the system requirements will be called by the project manager
- Members of the project team will create design solutions to the problems and write new system requirement tables
- The final design solution will be decided by the manager and all design documents and new system requirement tables will be peer reviewed by each member of the project team
- After the team reaches a consensus on the update, the system requirements can then be officially updated

The goal of the team should be to avoid this process at all costs by making good designs. If this process is required, the above process will ensure the project design remains functional.

2.4 House of Quality

The house of quality, shown in Table 4, is a tradeoff matrix that helped the team identify the relationship between different marketing and engineering requirements pertaining to the project. The marketing requirements listed below are the possible expectations/desires the customers have in mind for the project. The engineering requirements are the goals we would like to achieve for the project. Each requirement has an associated polarity in which depending on the symbol that was denoted for that requirement, it will increase the desirability of the product

Table 4: House of quality.

			Engineering Requirements								
			Power Output	Setup Time	Cost	Dimensions	Weight	Video Quality	Sensor Range	Accuracy	Signal Strength
			+	-	-	-	-	+	+	+	+
Marketing Requirements	Accuracy	+			↓			↑↑	↑↑	↑↑	↑
	Portability	+		↑	↓	↑	↑↑				
	Durability	+			↓	↓	↓				
	Usability	+			↓			↑		↑↑	
	Battery Life	+	↑		↓			↓	↓		↓
	Cost	-	↓		↑↑	↓	↓	↓↓	↓	↓↓	↓
	Install Ease	+	↓	↑↑	↓	↑	↑				
	Maintenance	-			↓	↑	↑				
	Area Coverage	+			↓			↑↑	↑	↑↑	↑
Targets for Engineering Requirements				< 20 min	< $200	< 10 x 10 x 10 in.	< 3 lbs		> 5 ft		

Legend

(+) Positive Polarity; (-) Negative Polarity; (↑) Positive Correlation; (↑↑) Strong Positive Correlation; (↓) Negative Correlation; (↓↓) Strong Negative Correlation

3.0 Research and Background Information

In order to design an effective product, the group must first do research on existing systems and technologies in order to find ways to improve them.

3.1 Existing Technologies

Many companies and buildings already have systems to monitor parking occupancy in their garages. This section will compare the advantages and disadvantages in these systems and discover ways for a new product to perform better.

3.1.1 Disney Springs

A highly functional parking garage monitoring system can be seen in parking garages at Walt Disney World's outdoor shopping complex Disney Springs. This system utilizes a camera above each spot to detect the presence of a vehicle. The parking availability information is displayed on a large sign at the entrance to the garage (as seen in Figure 1), as well as displayed on a small sign at the entrance to each floor. In addition to displaying the number of available spots, Disney's parking garages also have a small indicator light above each spot that is either green or red, informing the driver if the spot is available. While this system is highly accurate, the system does have some flaws, and requires a substantial amount of resources to implement. For an estimated 8000 spots in the parking garage, each spot requires a camera and an indicator light to operate. The parking garage roof was also built with the system in mind, as the roof of the garage has small pillars in front of every spot where the cameras can be mounted. The system also has difficulty detecting small vehicles such as motorcycles and smaller electric vehicles. Disney's system also does not display their parking availability information anywhere but on their physical signs.

Figure 1: Disney Springs parking garage signage.

3.1.2 RedStorm Parking Guidance System

Created by SignalTech, the RedStorm parking guidance system is a monitoring system that can be installed in pre-existing parking garages. RedStorm utilizes a "zone" concept, where instead of monitoring individual spots, the system instead tracks the number of cars in a current defined zone. The zones are defined in the parking garage by the customer, and the number of available parking spots are calculated before installation. The use of four differential zone counters and four infrared sensors (two for incoming, two for outgoing) at the entrance to each zone are used to track the number of cars between each zone. When a car enters or leaves a zone, the system will increment or decrement the number of available spots. The number of available spots is reported on digital signs at the entrance to the parking garage, and the customer also has the option to export the parking availability information as a .csv file for use by their web applications. All the infrared sensors on a floor are hardwired together in a daisy chain configuration, and a RS485 cable bus is used for communication. All the information is sent to a master controller which then handles updating the digital signs or exporting the data. The overall system described can be seen in the topology figure below (Figure 2). One of the interesting features of this system is the handling of non-vehicle traffic. When a vehicle enters the first and second infrared sensor's field of view, two relay outputs are triggered until the field of view returns to normal. The system then checks if both triggers are flipped for more than 300ms, which signifies that a vehicle has passed through, and to increment the vehicle count.

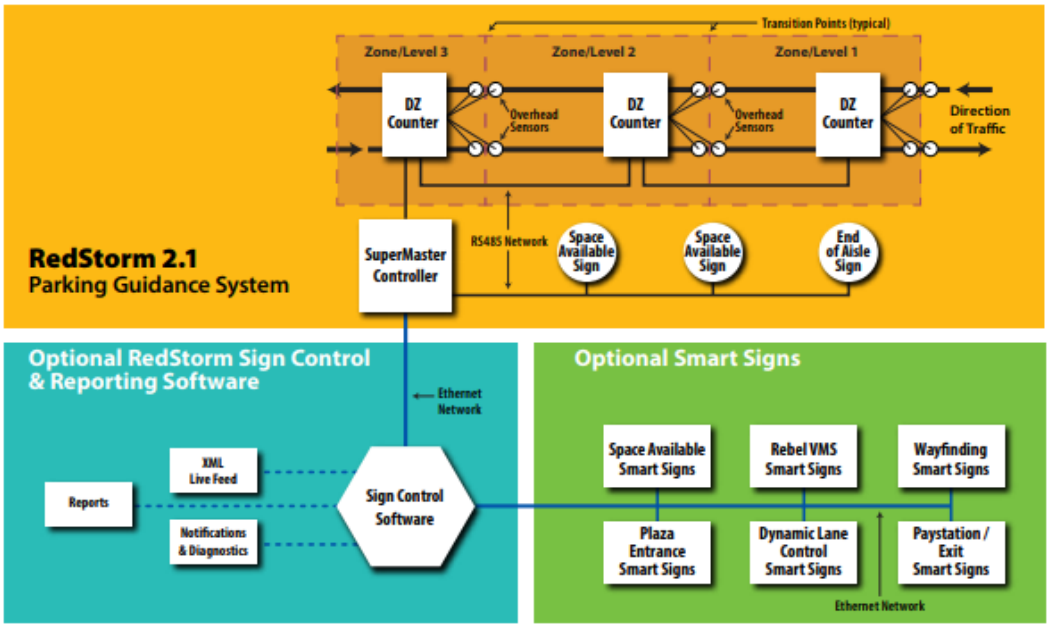


Figure 2: RedStorm system topology. Reprinted with permission from SignalTech.

3.1.3 Indect Parking Guidance System

The Indect parking guidance system has a multitude of solutions for monitoring parking, ranging from using individual cameras over every spot (such as Disney's

system) to using an electromagnetic sensor at the entrance to their garages. What separates Indect from other solutions is the use of a single camera to monitor multiple spots. With one camera mounted in between a row of spots, the camera can monitor up to six spots simultaneously. Indect's system has similar features to Disney's, with indicator lights above each spot and digital signage at the entrance to the garage and on each floor. Unlike both Disney and RedStorm, the Indect system takes the data gathered from their system such as peak hours and visit duration and displays this information on phone and web applications. The Indect system also includes special features, such as the ability for users to enter their license plate number on their phone and have the system search through all the vehicles and attempt to locate it. If it finds the vehicle a map displaying the location and directions to the user's car will be displayed. Like RedStorm, Indect's system can be installed in preexisting parking garages, with no modifications to the garages structure. The downside to using a multitude of detection methods is not only the cost of the different sensors, but the installation of an entire system becomes more complicated compared to a more straight-forward system such as RedStorm which uses primarily one type of detection.

3.2 Computer Vision Methods

For the purposes of this project, computer vision will be used to detect the presence or absence of a vehicle in a parking spot. In this section we will discuss some of the technologies available to perform computer vision related tasks, such as OpenCV and TensorFlow, and discuss possible methods for car detection.

OpenCV is an open-source library of programming functions that assist with performing real-time computer vision. This library is written in C++ but allows the use of other programming languages such as Python. Using OpenCV allows for multiple approaches for detecting space availability.

3.2.1 Python and OpenCV

Python is one of the most popular languages for use in both AI and computer vision and is quickly becoming standard use in both fields. Python allows for quick prototyping and is also a very easy to learn language. The language being easy to learn allows it to be very concise, which is ideal for our project where we must understand and implement complex concepts to accomplish our project goals. The language also supports a package installer called PIP, which allows for the installation of packages, such as OpenCV, incredibly straightforward with very little need for more in-depth installing methods.

3.2.1.1 Hough Transform Line Detection

The first step in a purely computer vision-based solution is to detect the number of individual parking spots. This not only avoids having to manually count the number of available spots but could also assist certain methods of detection as we would be able to divide the image into sections to perform our detection on. Hough Transforms is one possible method to accomplish this goal. A Hough Transform is a method for feature extraction in an image, in our case, the lines of a parking spot.

OpenCV includes functions, HoughLines and HoughLinesP, that will help accomplish this goal. Hough Transforms work by first scanning the image for edge points. Once it finds an edge point, it begins to scan the image for intersections with other edge points in the image. Points that have a higher amount of intersections can then be determined to be a line in the image and can then be marked. An example of the result from the Hough Transform run on an image of a parking lot can be seen in Figure 3.

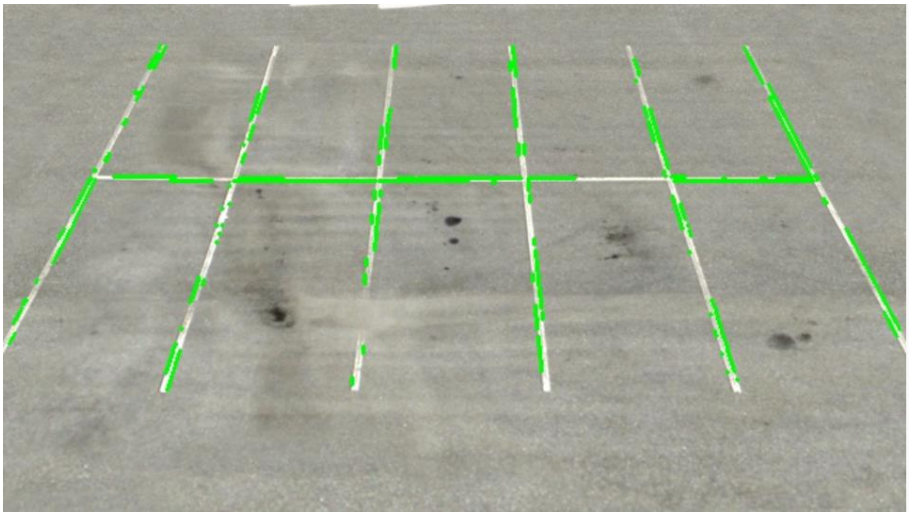


Figure 3: Hough Transform on parking lot.

3.2.1.2 Color Histogram Comparison

OpenCV has an included function, compareHist, that takes two histograms and return a numerical parameter that expresses how well those two histograms match each other. A possible, low computation solution to our detection problem would be to take a picture of a parking spot with no vehicle, create a color histogram, and compare that over time to the histogram of the live feed of an individual spot. If a car enters the parking spot, the value returned from the comparison of the two-color histograms would change drastically and allow us to update the availability of that spot.

3.2.1.3 Cascade Classifier for Haar Features

This detection method is based off detecting similar features in images, utilizing a Haar Feature. First defined by Viola and Jones, this method for object recognition uses a Haar Feature that looks similar to the desired object you are detecting. For example, a Haar Feature for a human face, we note that the eye region is considerably darker than the area surrounding the cheeks. The resulting Haar Feature used for detecting eyes can be seen in Figure 4.

The use of a cascade classifier for Haar features is used to tell OpenCV what exactly to look for in an image. Several classifiers for cars already exist and are open-source, however, the creation of a personal classifier is relatively straight forward. OpenCV includes both a trainer and a detector for Haar-cascade

detection. To create the classifier, we would take pictures of cars parked in garages around UCF and use them to train our classifier. With a set trained from this data, we can then run the detection method included in OpenCV to detect the available parking spots.

Figure 4: Haar Feature for eyes. Used under Creative Commons 4.0 International License.

3.2.1.4 HOG + Linear SVM

This method utilizes the concept of a Histogram of Oriented Gradients (HOG), in addition to the use of classifier algorithm called Linear SVM. The Histogram of Oriented Gradients method first groups pixels into small cells, and then calculates the gradient direction and magnitude. These values are then stored into bins, and the resulting histogram gives a representation of the orientation of that cell. Repeating this process for each cell gives us a representation of the structure of the image. An example of the output of the HOG method can be seen in Figure 5.

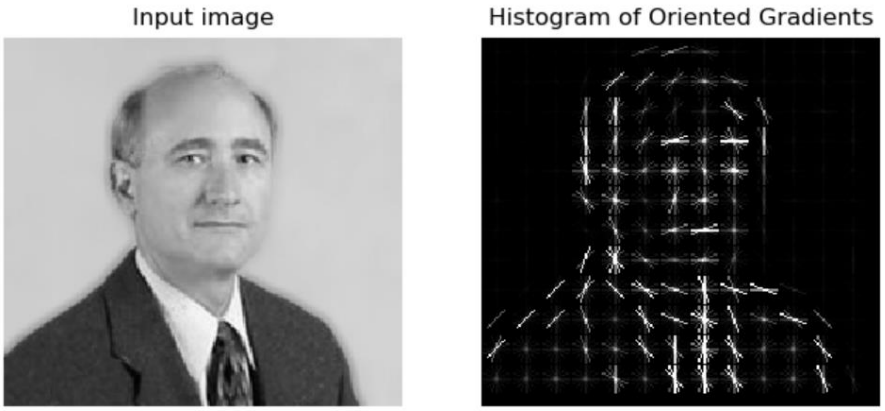


Figure 5: Histogram of Oriented Gradients output.

A Linear Support Vector Machine (SVM) is a classifier training method. The method works by taking two sets of data, positive and negative examples, and running the SVM algorithm and creating a model, which can then be used to classify new examples. For our purposes, we would extract HOG for images of parked cars and empty spots, and then use those to train a SVM classifier. OpenCV already includes support for both HOG and SVM in their libraries.

3.2.2 TensorFlow and Keras

TesnorFlow and Keras are both tools used extensively in Deep Learning and are both relatively new technologies. Both of these tools were released in 2015. TensorFlow, developed by Google, is an open-source library that is used for deep learning, machine learning, and neural network applications. Google supplies a very large API for use with TensorFlow, that includes object detection for cars. Tensorflow also gives the group the opportunity to train the system by supplying samples of cars in order to improve the detection. TensorFlow supplies APIs for use in different languages, with the most stable API being supplied for Python.

Keras is another technology used for deep learning. Keras can be used to accomplish a lot of the same tasks as TensorFlow. Keras provides a streamlined API which makes the use of Keras much easier compared to TensorFlow which is very appealing for Computer Vision beginners. Keras can also be used as a high-level abstraction for TensorFlow, allowing the use of TensorFlow libraries through Keras.

The use of these technologies would allow for very accurate detection of vehicles. However, running such a computationally expensive process would put a large recourse requirement and drain on our Raspberry Pi, with most benchmarks running TensorFlow and Keras achieving only 3 frames per second. Considerations such as running detection algorithms only on a fixed time interval or offloading computation to a cloud-based Internet service such as AWS can be considered. Testing will need to be done to determine if this method is possible for our system.

3.2.2.1 You Only Look Once (YOLO)

You Only Look Once, or YOLO, is an extremely new method used for object detection, that uses a Convolutional Neural Network. YOLO has outperformed other detection methods, such R-CNN, and uses a completely different approach for image detection. The method looks at an image once and divides the image into a grid of 13 by 13 cells with each cell being responsible for predicting 5 bounding boxes. This produces a total of 854 bounding boxes. It then produces a confidence score for each box and tells us if the bounding box is accurate. This confidence value is used to get rid of a large amount of the boxes created. It also attempts to predict the class of an object, trying to discover what the object within the box is, such as a car or dog. The output of running YOLO on an image can be seen in Figure 6. What makes YOLO so efficient is that all these boxes are created at the same time, hence the name You Only Look Once. Several implementations of YOLO have been created using both TensorFlow and Keras. For this project, the implementation called Yad2k will be used, which uses both TensorFlow and Keras to implement the YOLO method. The use of a less expensive version of YOLO is also available called Tiny YOLO. This implementation uses only 516 MB of memory and results in high frames on almost all devices. The tradeoff comes in the form of a somewhat lower level of accuracy and the speed of results.

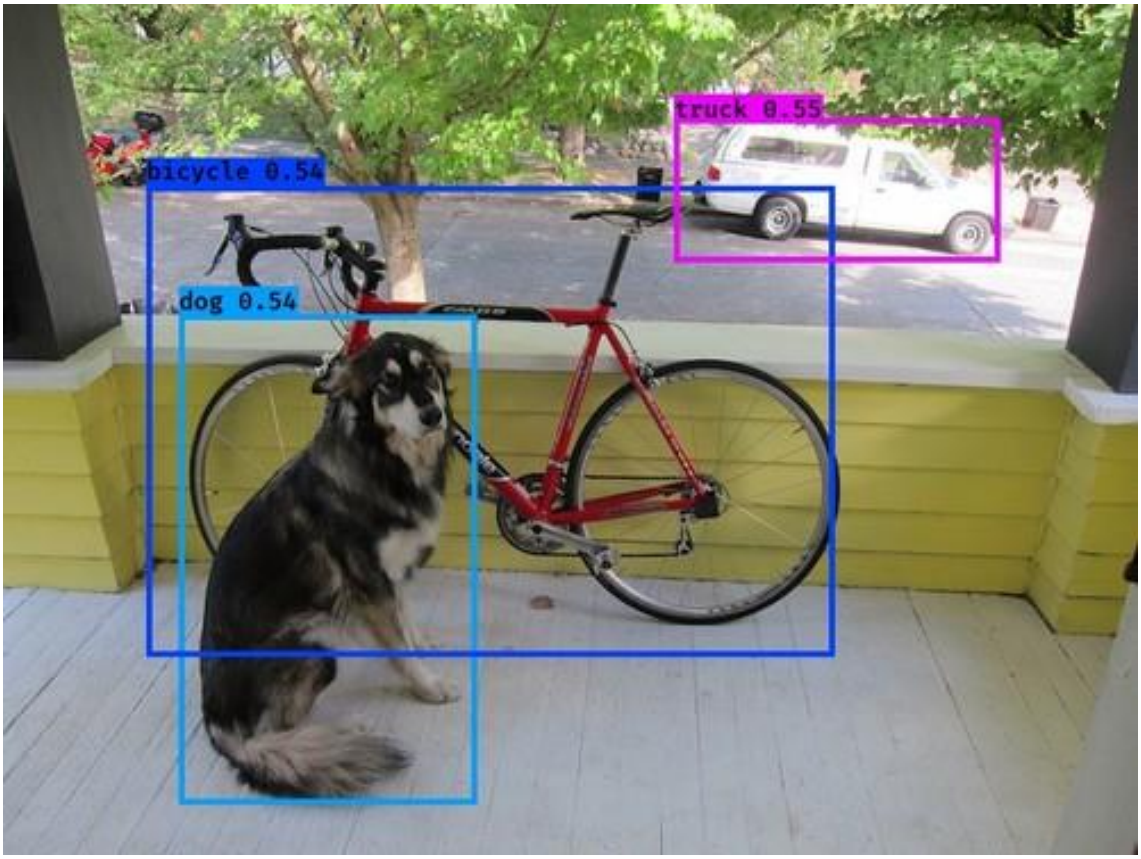


Figure 6: Example of YOLO output. Used under MIT license.

3.2.3 Comparison of Computer Vision Technologies

In this section we will compare the different object detection methods discussed in the above sections, and ultimately choose a method for our project.

The first detection method we discussed was the Color Histogram method, where we simply compare the histogram of an empty spot with the histogram of an occupied space. This method has an extremely low computational cost but suffers greatly in accuracy, as this method does not attempt to detect an object. Issues can also arise with certain colors of cars, being that a yellow car would cause a large change, but a gray car would appear like the parking garage background and therefore would not be detected.

The second method discussed was the Viola and Jones Haar Classifier method. This method works well with faces and objects that have similar features. However, this method suffers from the need to hard code the features we are looking for. For example, if we rotate the image of a face, the method has trouble detecting it, because the vector was defined horizontally. With the differences in car makes and models, this could pose an issue. This technology is also relatively old, first suggested in 2001, and more accurate and advanced technologies have been implemented.

The next method discussed was the Histogram of Oriented gradients, which built upon Viola and Jones method, and offers an updated and more accurate object detection method. This method, while more accurate than all previous methods, still relies on hard coding features, and does not take advantage of new technologies. This method was first suggested in 2005, and with the era of Deep Learning beginning in 2012, new methods and technologies have created more accurate and complex methods. For the purposes of this project, with limited computing power, this design could still accomplish our desired tasks.

The final method discussed is based off using very new technologies such as TensorFlow and Keras, as well as the use of OpenCV libraries, and is one of the most accurate detection methods available today, YOLO. This method, for the purposes of this project, would provide an extremely accurate object detection method for cars which would allow us to determine the availability of spots based off the number of cars present in a given image. The tradeoff is the computational power needed to run such a method. The complexity of this method would also require a significant amount of time implementing and testing.

The Tiny YOLO method briefly discussed in this section offers an alternative to YOLO and utilizes a smaller amount of memory and the price of a small drop in performance. These drops are still incredibly more accurate and efficient compared to the other methods however, and still comes with all the benefits of the full version of YOLO.

The following chart shows the main criteria used for comparison of these methods.

Table 5: Comparison of computer vision methods.

Methods	Complexity of Implementation	Computational Power Need	Accuracy
Color Historgram	Low	Low	Low
Haar Classifier	Medium	Low	Low
HOG + linear SVM	Medium	Low	Medium
YOLO	High	High	High
Tiny YOLO	High	Medium	High

In this section we have discussed numerous methods, all of which could be used to accomplish our requirements with varying degrees of accuracy. After comparing each method, and discussing the pros and cons of each, the **Tiny YOLO detection method** will be used. Given the amount of time we have been given to implement this system, and the accuracy that results from this method, our team is confident in the ability to utilize this method. Testing will need to be done to ensure this method can be run effectively on our system, as the power of the Raspberry Pi is limited. If this method is unable to run our system, the HOG + linear SVM method will be used, as the accuracy is still at an acceptable level.

3.3 Microcontroller Considerations

At the heart of each of the two sensor types (the camera sensor and the parking space sensor) in this project must be a CPU to control the devices and handle communication with other sensors and the main server. Due to the different power requirements and calculation complexity of the camera sensors versus the parking space sensors, each will use a different microcontroller. The camera system requires a relatively powerful controller that can handle image processing, which will require more power draw in addition to the camera itself. However, the wireless parking spot module, utilizing sensors, requires very little power and needs only extremely basic processing functionality, as the sensor will effectively report only a binary state: “on” or “off.” The bulk of the processing requirement will likely be used instead for handling communication with the rest of the parking occupancy monitoring system.

Therefore, the main criteria to be considered for the microcontroller is that it should as low of a power draw as possible by utilizing sleep modes. The ability for the sensor to spend the majority of its time in a low power sleep mode is what will allow the sensor to last a year or more on battery power alone. A frequent check of a parking spot status might be once every thirty seconds; and there does not need to be updates on this status as frequently when the garage is less full. As a result, the sleep cycle can be set even longer at certain times in the day, saving even more power.

3.3.1 Arduino Microcontrollers

Originating from the Ivrea Interaction Design Institute, Arduino boards are open-source platform that consists of both a microcontroller and an Integrated Development Environment (IDE). All Arduino boards are programmed through Arduino IDE, the software that allows one to write and upload computer code written in basic C++ to the board. An Arduino board would be used as the CPU for the sensor system and the camera system. The main benefit of Arduino is that due to its immense popularity there is a wealth of information and documentation on the internet about any situation one could face when designing a circuit with Arduino. The repository of knowledge on Arduino will make it easy for the group to learn the platform and to design a circuit around it. In this section, a variety of Arduino boards will be discussed.

3.3.1.1 Arduino Uno

The Arduino Uno is one of the most popular boards Arduino has available, especially because it is beginner friendly. It has 14 digital input/output pins, 6 analog inputs, a USB connection, a power jack, a reset button, an ICSP header, and a 16 MHz ceramic resonator. The Arduino Uno board does not use the FTDI USB-to-serial driver chip unlike preceding boards.

3.3.1.2 Arduino Mega

The Arduino Mega is like the bigger version of the Arduino Uno. It has 54 digital input/output pins, 16 analog inputs, a reset button, a power jack, a USB connection, a 16 MHz crystal oscillator, and an ICSP header. The large amount of

input/output pins available on this board makes it an attractive choice for projects that involve, for instance, a bunch of LEDs or buttons.

3.3.1.3 Arduino Leonardo

The Arduino Leonardo board has a built-in USB functionality available in the microcontroller unlike preceding Arduino boards. This makes it possible for the board to appear as a mouse or keyboard. It has 23 digital input/output pins, 12 analog inputs, a 16 MHz crystal oscillator, a micro USB connection, a power jack, an ICSP header, and a reset button.

3.3.1.4 Board Specifications

All three Arduino boards share the same input voltage value range (7-12V) and the same clock speed (16 MHz). The following table was made to compare the different specifications each Arduino board has to offer.

Table 6: Comparison of Arduino variants.

Arduino Board	Microcontroller	Flash Memory	SRAM	Digital I/O Pins	Dimensions
Arduino Uno	ATmega328	32 KB	2 KB	14	2.7” x 2.1”
Arduino Mega	ATmega2560	256 KB	8 KB	54	4” x 2.1”
Arduino Leonardo	ATmega32U4	32 KB	2.5 KB	23	2.7” x 2.1”

The main difference between the three models is the Microcontroller CPU used. This is the important part of the comparison, because if Arduino is chosen as the development platform for the wireless sensor, the Arduino itself will not be a part of the actual design. It will be used primarily for learning and development, as the final design will use only the ATmega CPU from the Arduino, and the rest of the circuit board will be designed by the group. The Arduino’s purpose other than ease of use and education is to be an example of what the ATmega CPU at its core is capable of; its functions can be replicated in an end-product designed circuit board. Because of the low processing requirement of the wireless sensor module, the preferable choice is to pick the weakest microcontroller of the three. In the case of ATmega, the difference between the three is the amount of memory available and the amount of input/output pins. Because the sensor module is ideally only going to use one sensor, there is no need for the higher pin counts. Even if a sensor fusion approach is used, most sensors will only use one or two pins each, making even 14 pins plenty. Out of the Arduino and ATmega CPUs, the ATmega328 is the most attractive option due to its lower price and power requirement, as well as its ease of installation as it has far fewer pins than the others.

3.3.2 Texas Instruments Microcontrollers

A microcontroller with low enough power consumption would allow a simple passive sensor to run solely from battery power, vastly simplifying installation

difficulty and cost of the parking system into any area, as line power infrastructure would not be required for each sensor to operate. The MSP430 microcontroller, manufactured by Texas Instruments, is specifically designed for ultra-low power consumption. Therefore, it is ideal to use in sensors such as the dedicated parking space sensors used by this project. The MSP430 has been produced for years; below are the relevant specifications for a current revision:

Table 7: Specifications for MSP430FR2000.

Specifications	Values
Price	$1.18
CPU	16-bit RISC architecture up to 16 MHz
Serial Communication	UART, IrDA, SPI
Analog	8-Channel 10-Bit ADC, Comparator
Input Voltage	1.8 V – 3.6 V
Power Consumption (Low Power Modes)	Active: 120uA/MHz Standby Mode: 1 uA Shutdown Mode: 34nA
GPIO	12 input/output pins 8 interrupt pins (can wake from low power modes)
Clock Sources	On-Chip 32-kHz RC Oscillator On-Chip 16-MHz Digitally Controlled Oscillator On-Chip 10-kHz Oscillator (Very Low Frequency) On-Chip High Frequency Modulation oscillator External 32-kHz Crystal Oscillator Configurable pre-scaler between 1 and 128

The Low Power Modes and built in Comparator with Programmable Hysteresis makes this MSP430 a very attractive option for controlling the wireless parking space sensor modules, because this is built-in ability to quickly compare a sensor input value to a definable threshold. Additionally, the MSP430 has another benefit: prior knowledge. While the main benefit of the Arduino platform is that it has a considerable knowledge base online, the project group already has experience with the MSP430 platform as it is the microcontroller that UCF coursework in the Electrical Engineering and Computer Engineering majors uses in the laboratory class sections. This means that there would be much less time required for the group to learn the platform, as they have already used the development environment tools and have already had experience in flashing programs onto the chip as well as connecting and operating peripherals through the input/output pins.

The biggest feature to note is the low power modes. But, in order to put it in perspective, the power draw must be compared to the power draw from the Arduino.

There are several variants of the MSP430 platform: most of them concern features built into the development board such as LED digit displays that are not useful for the parking spot detection modules. Therefore, we feel that we can use the lower end versions of the MSP430 with less RAM due to the simplicity of the programming involved in needing to track only a handful of sensors and wireless communication. Other editions, such as MSP430FR5994, are designed specifically for Ultrasonic sensing applications, which will be useful for the ultrasonic sensing version of the parking spot module.

3.3.3 Microcontrollers Within Bluetooth Transceiver Chips

While searching for the ideal microcontroller to use in the wireless sensing unit, the group also did research on which Bluetooth transceiver chip to use in parallel with the microcontroller. In doing so, the group found that the majority of the currently produced Bluetooth transceivers contain their own microcontrollers, built directly into the chip. Many are even more powerful than the ATmega and MSP430 and even utilize 32-bit architectures with floating point functionality. The wireless sensor module has extremely low processing power requirements; it only needs to wake up occasionally to poll the attached sensor at an interval of 30 seconds or longer depending on the chosen settings and desired update rate. The group decided that there is simply no reasonable need to have a separate connected microcontroller to manage Bluetooth and sensor operations when the attached Bluetooth microcontroller can take over all of the functionality. Therefore, **neither the ATmega nor the MSP430 were chosen**, and a separate microcontroller will not be part of the wireless sensor module's design. Research and considerations on which Bluetooth transceiver IC (and included microcontroller) will be used within the design are continued in a later section (Section 3.7).

3.4 Sensor Considerations

Due to the camera's limited scanning range, some parking spots will be in the camera's blind spot. To solve this issue, several types of sensors were taken into consideration that could potentially be used for those parking spots that are not in the camera's field of vision. For the sensor system, a sensor that provides low power consumption and a large sensing range is desired. In this section, four distinct kinds of sensors will be discussed: Hall effect sensors, infrared sensors, ultrasonic sensors, and inductive proximity sensors.

3.4.1 Hall Effect Sensor

A Hall effect sensor is known as a magnetic field sensor. The output voltage of the Hall effect sensor varies depending on the magnetic field the sensor is introduced to. A basic Hall effect sensor is made up of three components: a regulator, a differential amplifier, and the Hall element. The schematic of a basic Hall effect sensor is shown in Figure 7. The regulator is used to regulate the input voltage entering the sensor. The Hall element is a thin strip of semiconductor material

through which the current applied to the sensor will travel through. The differential amplifier is used to amplify the potential difference across the Hall element known as the Hall voltage. The Hall voltage is proportional to the strength of the magnetic field. If no magnetic field is present, then the output voltage of the sensor will simply be zero because there is no potential difference. The benefits of Hall effect sensors include low power consumption and cost effectiveness.

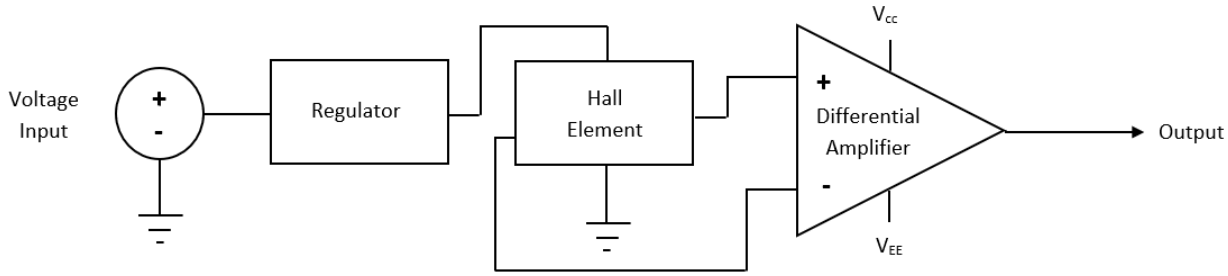


Figure 7: Basic Hall effect sensor schematic.

3.4.1.1 Analog and Digital Output Sensors

There are two main types of Hall effect sensors: analog output sensors and digital output sensors.

Analog output sensors, shown in Figure 8, are known for providing a continuous linear output voltage. The output voltage of an analog/linear output sensor is proportional to the present magnetic field. The Hall voltage can be calculated by using the following formula in which R_H represents the Hall effect co-efficient, I represents the current flow in amps, t represents the thickness of the sensor in mm, and B represents the magnetic flux density in Tesla:

$$V_H = R_H \left(\frac{I}{t} \times B \right)$$

The output voltage depends on the strength of the magnetic field. For example, if the strength of the magnetic field were increased, then the output voltage would increase until it reaches saturation. Also, depending on the polarity of the magnetic field, the output voltage will be either positive or negative. Both polarities and the output voltage response for each case can be seen in Figure 9.

The digital output sensor, shown in Figure 10, is similar to the analog output sensor, except the digital sensor has an additional component, a Schmitt trigger. The Schmitt trigger enables the digital sensor to switch between two states: “on” and “off”. The output voltage for a digital sensor depends on the sensor’s state. The Schmitt trigger determines the state of the sensor by comparing the output of the amplifier to the preset value of the sensor. The Schmitt trigger can switch the state of the sensor with ease due to the hysteresis loop generated by the trigger as shown in Figure 11. When the output of the amplifier is higher than the pre-set value, the sensor will be activated and set to its “on” state. When it’s less than the pre-set value, the sensor will turn off and there will be no output.

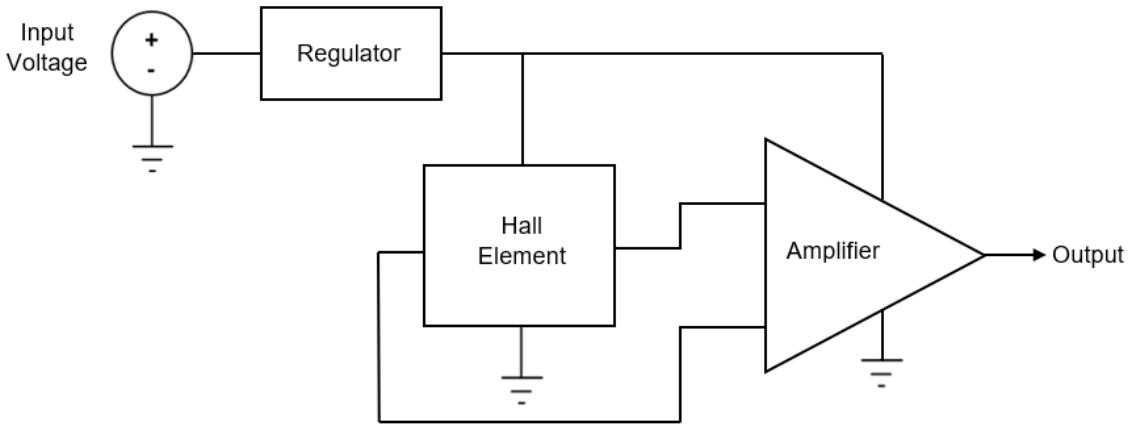


Figure 8: Analog Hall effect sensor schematic.

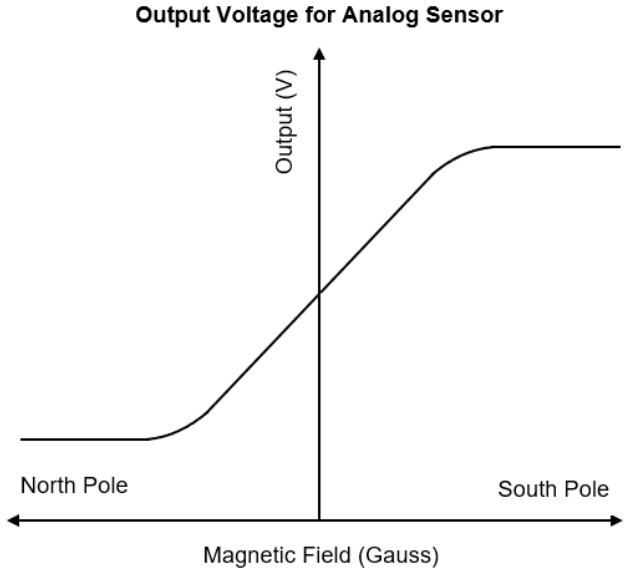


Figure 9: Output voltage vs. magnetic field plot for analog sensor.

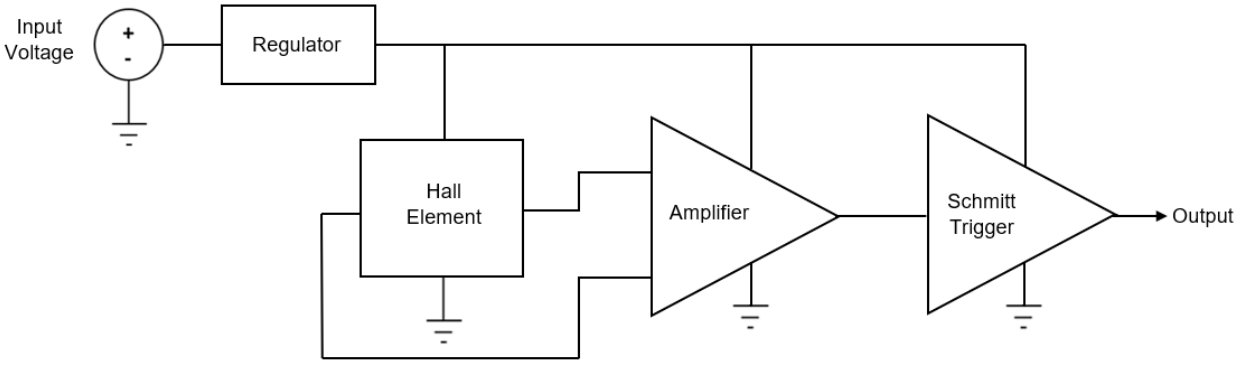


Figure 10: Digital Hall effect sensor schematic.

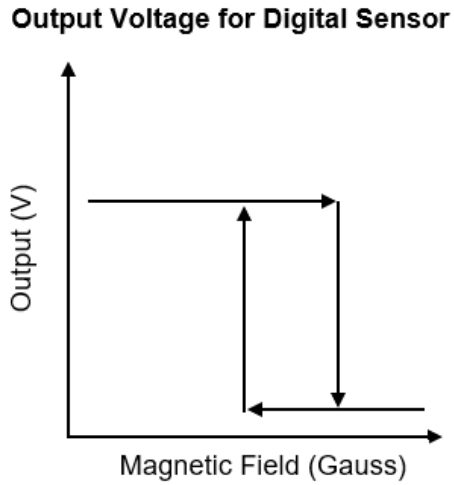


Figure 11: Output voltage vs. magnetic field plot for digital sensor with hysteresis loop.

3.4.1.2 Advantages and Disadvantages

Hall Effect Sensor Advantages

1. Ambient conditions such as dust, humidity, and vibrations do not affect Hall effect sensors. This is good news since the sensors will be placed in a parking garage, an environment that consists of all three previously mentioned conditions.
2. There is no friction or wear due to its ability to operate without any physical contact. Thus, its quality will be maintained over time and extends its lifespan.
3. Digital Hall effect sensors have an “on” and “off” state. While no vehicle is present, there will be less power consumption because the sensor will be in its “off” state.
4. Hall effect sensors can operate in a wide temperature range.
5. Hall effect sensors can switch without bounce because it does not use a mechanical contact. It can provide an immediate response when it detects an object.

Hall Effect Sensor Disadvantages

1. Hall effect sensors have a limited sensing range because magnetic sensing is only effective at short distances.
2. Since Hall effect sensors detect magnetic fields, it’s possible for other magnetic fields can cause interference. The sensor could be activated by a different source instead of a vehicle causing the sensor to deliver false information.
3. The sensitivity of Hall effect sensors is affected by temperature.
4. When there are physical inaccuracies, an offset voltage could appear as an output voltage when no magnetic field is present.

The main disadvantage that makes Hall effect sensors a bit undesirable is the limited sensing range it offers. A larger sensing range might be required to detect the vehicle in the parking space. The idea was to place a sensor unit on the floor in a parking spot with the head of the sensor pointing upward. When the spot becomes occupied by a vehicle, the sensor would detect the bottom part of the vehicle. The distance between the ceiling and the roof of a vehicle is at least 10 feet, depending on the vehicle model, which is much larger than a few inches. The same situation is encountered if the sensor were placed on the wall because the distance between the wall and the front of vehicle, depending how the person parked their vehicle, is more than few inches.

3.4.2 Infrared Sensor

Another technology that may be considered for the sensor system is infrared sensors. Infrared is a frequency of light waves that is outside the range of human visibility. It is light in the wavelength range between 700 nm to 1000 nm.

Infrared is a low frequency wave meaning it has a very low penetration and will not pass through most walls or objects. Because of this behavior it has many practical applications especially in the field of detection. In this section, two types of infrared sensors will be considered: passive infrared sensors and active infrared sensors.

3.4.2.1 Passive Infrared Sensor

A passive infrared (PIR) sensor is a device that measures the amount of infrared light entering it. Because it is a passive sensor, it does not emit any infrared light, instead it will detect the infrared radiating from heat sources such as the human body. Therefore, its most practical use is detecting changes in the surrounding environment. For example, if an object moves in front of the PIR and blocks it, the amount of infrared light received by the sensor will temporarily change. It may be reduced as the object blocks infrared light from entering the PIR, or it may increase as the object moving in front of the PIR may be radiating more infrared than the average of the surrounding environment. The PIR will compare this change to an internal threshold and will trigger an output if the change is greater.

The PIR can be used to detect moving vehicles, as vehicles may drive over the sensor and block incoming infrared light, enabling the PIR to be used as a parking status indicator. However, vehicles also emit infrared light because it's heat source, especially the underside of vehicles where the exhaust system is located. After the car is parked and shut down, it will cool off over time and emit less infrared light. Therefore, the vehicle will not be a consistent source of IR light which may affect the usefulness of PIR sensors unless carefully tuned for these situations.

3.4.2.2 Active Infrared Sensor

Another form of IR sensor is a simple active infrared sensor which emits IR light from a transmitter into a receiver. This will form a constant line of IR between them. Because IR does not penetrate most objects, the receiver will not receive the IR beam if an object comes between the transmitter and the receiver. This means that IR can be used as a trip sensor, which means that an output will be generated

if an object “trips” the IR beam by blocking the wave and preventing it from entering the receiver.

This is another practical method for detecting vehicles. If a vehicle enters the IR beam, it will generate an output indicating the parking spot is occupied. However, there are several drawbacks for this application. It is an active infrared sensor, which means it will constantly be consuming power to continuously emit the IR light into the receiver. This means it is not very ideal for a system targeting the lowest power consumption possible. Additionally, the system requires a separate receiver to be some distance away (such as the ceiling), which will have to be connected back to the sensor system. In some places, such as an open parking lot with no overhead roof, there is no place for the receiver sensor to go unless overhead poles are installed for each parking spot. Additionally, the precise placement of the transmitter and receiver may mean that smaller vehicles such as smart cars or motorcycles may not enter the IR beam and not set off the sensor, even though the parking spot is occupied.

3.4.2.3 Advantages and Disadvantages

Infrared Sensor Advantages

1. Infrared sensors offer a sensing range of around 10 meters. This exceeds the amount necessary to achieve the sensor system’s purpose.
2. Active infrared sensors can reliably detect objects in a low-light setting.
3. Infrared sensors work in real time. It would be able to detect a vehicle moving almost instantly.

Infrared Sensor Disadvantages

1. Infrared light cannot travel through solid objects such as walls or doors.
2. A line of sight between a transmitter and receiver is required. Anything that comes in between these two components can trigger the sensor. This means that not only the vehicle can activate the sensor, other objects can trigger the sensor as well. This could relay inaccurate information.
3. Infrared sensing is based on temperature. The sensor cannot tell the distinguish the difference between two objects that have similar temperature range. This would alter the results of the sensor reading.
4. Moisture and humidity decreases the sensing reliability of the infrared sensor reading.

The main disadvantage with using infrared sensors is that it’s based on temperature. Infrared sensors can detect infrared radiating from heat sources. A vehicle could be considered a heat source after it’s been running for a while. When the vehicle parks in a spot, the sensor would detect the vehicle due to the elevated temperature it’s emitting. But as time passes by, eventually the vehicle’s temperature would drop down to a much lower temperature compared to its initial temperature. The vehicle isn’t a consistent source of heat source, so it could affect the infrared sensor output. Since humans are also a source of heat, infrared

sensors can detect their presence. The infrared sensor would mark the parking space as occupied, thus, relaying false information because it can't tell the difference between a person and a vehicle.

3.4.3 Ultrasonic Sensor

Another method for vehicle detection would be ultrasonic sensors. Ultrasonic sensors can measure distance using ultrasonic waves. Ultrasonic waves are sound waves of frequencies higher than the maximum audible limit for people meaning we cannot hear them (above 20 KHZ). The sensor consists of an ultrasonic transmitter and an ultrasonic receiver. The sensor can measure distance by emitting ultrasonic waves at a certain frequency and retrieving the reflected sound wave as shown in Figure 12.

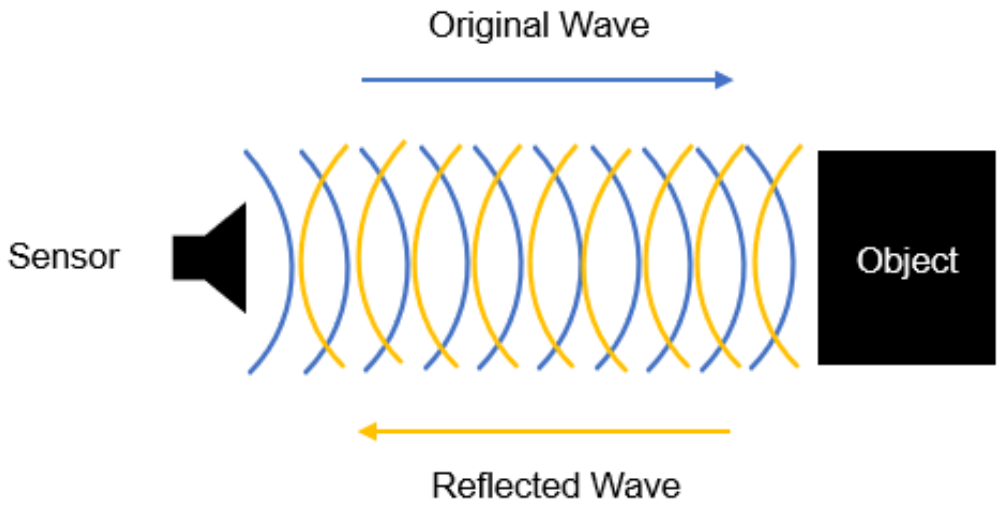


Figure 12: Theoretical demonstration of ultrasonic sensor.

The distance between the sensor and the object can be calculated by recording the amount of time it took for the wave to bounce back to sensor. The following formula can be used to calculate the distance, knowing that the speed of sound is 344 m/s at 20° C.

$$\text{Distance} = \frac{\text{Time Taken} \times \text{Speed of Sound}}{2}$$

3.4.3.1 Advantages and Disadvantages

Ultrasonic Sensor Advantages

1. Unless there's a heavy build-up, certain environmental conditions such as dust, humidity, and dirt do not affect ultrasonic sensors. This is good news since the sensors will be placed in a parking garage, an environment that consists such conditions.

2. Ultrasonic sensors have a higher sensing range (about 10 meters) compared to most sensors meaning it can be placed on the ceiling or wall and still be able to detect the vehicle occupying the parking space.
3. Detection of objects by ultrasonic sensors is unaffected by its color, transparency, or optical reflectivity.

Ultrasonic Sensor Disadvantages

1. The size, shape, and angle of an object can affect the accuracy of the ultrasonic sensor's readings. Ideally, the surface of the target would be hard and smooth.
2. Ultrasonic sensors are sensitive to variation in temperature. It affects the accuracy of the sensor.
3. It's difficult for ultrasonic sensors to detect objects of low density, such as fabric or foam, because they absorb sound waves. It decreases the sensor's accuracy.

The main disadvantage of ultrasonic sensors is that there's a possibility that some objects won't be detected by the sensor. An object's shape or position could deflect the ultrasonic wave away instead of reflecting it back to the sensor. Small objects are difficult to detect because they might not have the surface area necessary to reflect enough of the ultrasonic wave back. If the sensor is placed on the ceiling, this issue shouldn't have too big of an impact on our project because the ultrasonic waves would be colliding with the roof of the vehicle. The roof of a car has a large enough surface area and it's a hard, smooth, and slightly leveled surface. The sensor should also be able to detect motorcycles due to its size, although the reading might not be accurate due to its uneven surface. Some objects can absorb the ultrasonic wave, such as carpeting, making it almost impossible to detect. Fortunately, for this project, vehicles are made of metal and metal is known to reflect approximately 100% of ultrasonic waves making it easy to detect.

3.4.4 Inductive Proximity Sensor

An inductive proximity sensor can detect metallic objects without coming into contact with the object. The sensing range for inductive proximity sensors is up to 60 mm. The operating distance depends on the sensor's shape and size. The main components of an inductive proximity sensor are an oscillator, an inductive coil, a capacitor, a detector, an output circuit, and an energy source. The oscillator generates an electromagnetic field out of the face of the sensor. When a metallic object approaches the electromagnetic field created by the sensor, eddy currents are generated on the object's surface. Once the metallic object enters the electromagnetic field, the eddy currents at the surface of the object will increase and cause the amplitude of the oscillation to decrease. Eventually, the oscillation will decrease to the point that it surpasses the threshold. The Schmitt trigger in the circuit will notice when it has gone below the threshold. Once the Schmitt trigger in the sensor senses this change, the sensor will be activated and indicate the presence of a metallic object. When the metallic object exits the magnetic field, the sensor will switch back off and return to its original state.

3.4.4.1 Advantages and Disadvantages

Inductive Proximity Sensor Advantages

1. Inductive proximity sensors can detect metallic objects without contact. Thus, its quality will be maintained over time and extends its lifespan.
2. Inductive proximity sensors can only detect metallic objects. This limits the risk of the sensor being triggered by non-metallic objects such as humans or plastic debris.
3. These sensors can operate in harsh environmental conditions meaning the environmental conditions in a parking garage won't have an influence on the sensor.

Inductive Proximity Sensor Disadvantages

1. Inductive proximity sensors have an extremely limited operating range, an average of about 60 mm.
2. The sensing range of the sensor is affected by its size, its shape, the coil size, and the type of metal the object is made of.

One the disadvantages of an inductive proximity sensor is that the sensing range of the sensor is affected by the metal the target is made of. Depending on the type of metal the object inside the electromagnetic field is made of, the induction level will differ. For instance, inductive proximity sensors best detect ferrous metal. Non-ferrous metal can be detected, but with a reduced sensing range. Each type of metal will possess a correction/reduction factor used to calculate the amount by which the metal will reduce the sensing range. For example, if a proximity sensor had a sensing range of 10 mm and metal the object is made of had a correction factor of 0.3, then the object will decrease the sensing range to between 3 mm.

Typically, car frames and their chassis are composed of carbon steel. If this is true, the sensing range for the inductive proximity sensor won't be affected as much since carbon steel is a ferrous metal. Unfortunately, the sensing range of an inductive proximity sensor is small to begin with. Also, the thicker the material, the harder it is for the sensor to detect the object because the eddy currents will be dispersed by it.

3.4.5 Sensor Comparison and Selection

After researching and considering the advantages and disadvantages for each sensor, the final selection comes down to some crucial key points. One key factor is the sensor's sensing distance, in other words its sensing range. For the sensor system, the sensor should offer a sensing distance of at least 10 feet (3 meters). Whether the sensor is placed on the ceiling or on the wall, 10 feet should provide enough sensing distance for the sensor to detect the vehicle. Another key factor is low power consumption. Ideally, the supply voltage for the sensor shouldn't surpass 5 or 6 volts. An additional factor is the operating temperature range of the sensor. The sensor should be able to handle a wide temperature range because the system will be exposed to variant temperatures outside in a parking garage.

The sensor should also be able to handle certain environmental conditions that it will be exposed to in a parking garage such dust, dirt, and humidity. The sensor should also have a long lifespan. We wouldn't want a sensor that lasts for only a few months or a year. That wouldn't be efficient. Preferably, the sensor should have a quick response time when it comes to detecting an object. We're aiming to update the status of the parking spaces every few seconds, so using a sensor with a slow response time wouldn't allow us to do so. Lastly, the cost of the sensor shouldn't exceed twenty dollars. If the price of the sensor were high to begin with, then the cost of the sensor system would sky rocket since each parking space would require a sensor unit. Such an expensive system wouldn't be marketable or appealing to the customer.

Listed below in Table 8 are the specifications for the four types of sensors that were previously discussed in this section. Manufacturers offer such a wide selection of sensors with differing features to satisfy customer demands that it's difficult to jot down an average value for each of their specifications. Thus, to conduct an appropriate comparison, a specific sensor product was chosen for each type. The sensor products listed below are a DRV5033 digital-omnipolar-switch hall effect sensor, an HC-SR04 ultrasonic sensor, a GP2Y0A02YK0F infrared proximity sensor, and a DW-AD-511-M30 inductive proximity sensor. The specifications taken into consideration for each sensor were their maximum sensing distance, the supply voltage required to power them, their operating temperature, and their price.

Table 8: Comparison of sensor specifications.

Specifications	Hall Effect	Ultrasonic	Infrared	Inductive
Max Sensing Distance	12 mT*	4 m	1.5 m	0.04 m
Supply Voltage	2.5 to 38V	4.5 to 5.5V	4.5 to 5.5V	10 to 30V
Operating Temperature	-40 to 125℃	-15 to 70℃	-10 to 60℃	-25 to 75℃
Cost	$0.83	$3.95	$14.95	$62

*Hall effect sensors are triggered by a change in the magnetic field. Instead of a sensing distance, its maximum operating point was written.

After comparing each sensor type, while keeping the key factors in mind, a final decision was made. Group D decided to use **ultrasonic sensors** as the main component for our sensor system. Compared to all the other sensors, ultrasonic sensors offer the most sensing distance. The range it offers, 4 meters, is more than enough for our design purposes. Ultrasonic sensors can also operate under the temperature range and environmental conditions present in a parking garage. The ultrasonic sensor doesn't require a high voltage supply to operate, therefore, providing low power consumption. The cost of the ultrasonic sensor falls within our budget. The low cost makes it possible to create a sensor unit that doesn't exceed the budget amount for each parking space. The purpose of this project is to detect

the presence of a car, not the exact distance of an object. Therefore, even though the accuracy of the measurement is affected by certain factors, it is not essential to our design.

3.5 Camera

The Park Shark system will be using multiple cameras to implement CV (computer vision). They will be located on each side of the parking garage. They are an important aspect to the system because of their ability to cover multiple cars at once. The camera will be in communication with the base system. The base system will then take care of any incoming data, process the data, then output the results.

Our system will be using the same cameras to keep installation and maintenance simple. Having one camera makes it easier on the builder of the system and on the user of the system because you will only have to understand one camera instead of two or more. Another great perk of having one kind of camera is not having as much trouble with incompatible parts. Keeping the system simple will help economically because of less time installing, and less problems to deal with. The camera being chosen should work well with our base system and the ability to implement CV accurately. The most important part for the camera should be its ability to use CV to understand when a car is using a parking space or not. Another significant aspect but not as important is its ability to communicate with the base system.

3.5.1 Camera Options

There is a large variety of cameras to choose from. Most of the team was unfamiliar with the availability of cameras that could be used for our specific wants. That being the ability to us CV with accuracy, and its compatibility with our potential base system. This lead us to discuss our dilemma with Dr. Richie we were informed about the Pixy Smart Vision Sensor, which seemed like a good fit. After doing some research we came up with a few other options that seemed like good fit for our project. Those options being the AruCAM USB Camera Shield, SainSmart Surveillance Camera, OV7670 Camera Module by Atomic Market, and Arducam 5 MP OV5647.

3.5.1.1 Pixy Smart Vision Sensor

The Pixy Smart Vision Sensor (as seen in Figure 13) was first release back in March of 2014, and has since then become a popular use for projects using CV. The Pixy Smart Vision Sensor has an incredibly number of perks that come with it. The greatest perks of all is the fact that it comes with the ability to implement CV without having to use any loop holes to use OpenCV on some MCU. This devices ability to implement CV comes with some limitations. For example, it does not support the ability for face recognition, but in our case, this doesn’t really matter. Another constraint is how this device detects objects of interest. It uses a hue-based color filtering algorithm to detect objects, so it will not work well for object that are black, white or gray. There have also been some remarks about how the devices does not handle poor lighting well. So, lighting conditions should be taken

into consideration. The most impactful disadvantage to this camera is that it does not allow for users to use their own CV techniques. This keeps the project simple, which is a great advantage, but also takes away from the ability to further improve the camera's accuracy on car detection which is the most important job of the camera selected.

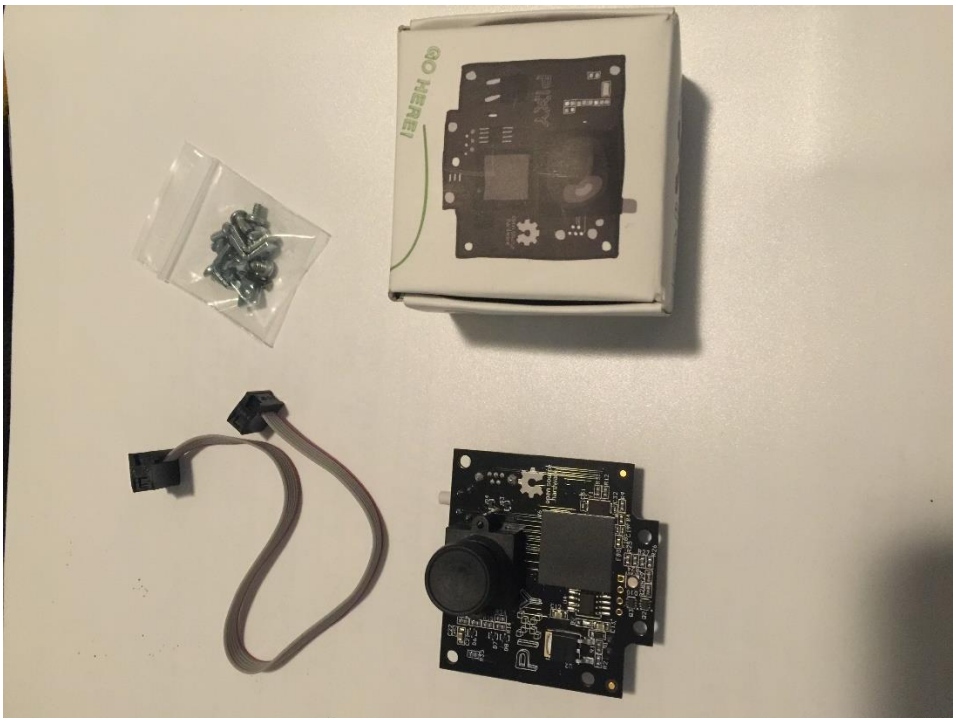


Figure 13: Pixy Smart Vision Sensor.

Some other great things about the Pixy Smart Vision Sensor is that it is small, fast, and is easy to use. The dimensions are 1.5 x 2.25 x 2.25 inch and weighs about 1.6 ounces. It has the ability to output what it detects 50 times per second, which is done by processing an entire frame in 20 ms. Though it is very fast it can result in more frequent false positive results. Results that believing an object is detected but really, it’s something else. As well as resulting in false negatives where it fails to detect an object of interest. The great thing is that you can improve detection accuracy by adjusting some settings based on what specifications your project is looking for. If you are dealing with issues of false positive and false negative you can adjust the signature tuning. The issue of poor lighting which was mentioned earlier can be addressed in different ways. When an object is overexposed because of too much light you can enable the overexposure highlighting and adjust the cameras brightness/exposure. This option allows you to visually see what parts of your object is overexposed and allows you to adjust easily. Another way to handle lighting issues is by adjusting the minimum brightness. This is an issue when dealing with darker colors, which results in some pixels not be considered as part of the object because they are not considered as part of any color signature. The camera is very easy to use with the Arduino, which is a consideration for our base system. The Pixy Smart Vision Sensor comes with a cable to plug right into

the Arduino, which allows for a quick start. If we are not able to use the Arduino for whatever reason the camera is compatible with many other options like Raspberry Pi, BeagleBone Black, and much more. It is also easy to use because it avoids the issue of using OpenCV on an Arduino. This is an issue because OpenCV is not portable to microcontroller systems. There are alternatives like OpenVx, but because the Pixy Smart Vision Sensor comes with CV it avoids all that trouble. Another simplicity comes in teaching the camera an object. All you do is press a button. There is a bit more to it. When you hold the button, an LED will turn on and will go through a pattern of colors. Each LED color is mapped to the number of color signatures you can teach the camera. So, once you hold the button until the LED color you want, you then have to hold an object in front of the camera and it will take care of the rest. It locks onto the object and determines what pixels are a part of the object and which ones are a part of the background. You can then determine if it made a good match by comparing the current LED color to the object. If it is a close resemblance, then you have a good match.

The Pixy Smart Vision Sensor can be bought and used at the time being for $69.00 each. This may seem like a big price but with everything that it brings the price is more than reasonable. It comes with the NXP LPPC4330 processor, 204 MHz, and dual core. The camera can detect an object that covers 4x1 pixels in the image. It also has a field-of-view of 75 degrees horizontal and 47 degrees vertical. This is the lens that is originally on the camera when shipped, but it can be replaced with a lens with longer focal length to increase the detection distance. The team has decided to keep the original lens, if chosen, to keep the idea of simplicity. The camera has a latency of 15ms and an update rate of 50 Hz. When it comes to powering this camera, there are a few options. The options being power through USB cable/connector (regulated 5V), power through the I/O connector (regulated 5V), and power through the power connector (unregulated 6V to 10 V). The first two options would be used if dealing with the Arduino because when we are setting up the device we will have the Arduino cable and the USB cable plugged into the camera simultaneously. The producers of Pixy Smart Vision Sensor assure that having both at the same time is not a problem. Something to consider when dealing with the I/O connector is that some of the pins are not reverse-polarity protect, so you could destroy the camera. The third option will not be used. This camera has a typical power consumption of 140 mA at 5V. This camera comes many online resources which describe in detail how to set up the camera, how to camera works, and even example projects to better understand it.

3.5.1.2 SainSmart Surveillance Camera

The SainSmart Surveillance Camera was chosen to be researched because of its compatibility with the Arduino, which is being considered for our base system. As well as seeing it considered as a top choice when using an Arduino. The SainSmart Surveillance Camera can be bought and used at the time being for $25.99 each.

This camera features MIPI (Mobile Industry Processor Interface) and CSI (Camera Serial Interface). This camera can handle a large number of pixels to be used by the processor. It can support 1080p at 30 fps, 720p at 60 fps and 640 x 480p video

recording. The camera sensor has a native resolution of 5MP and has a fixed focus lens. It comes with an image resolution of 2592 x 1944. The camera as comes with some undesired specifications like image color of black and white only.

With the ability to capture 720p at 60 fps, having a resolution of 5MP, and various other features, making it a great camera. It still comes with its downfalls especially when it comes to our objective in this project. Which would require us to use CV with only black and white images. Along with its lack of online resources. The camera seems to support things we do not really need and lacks things we are looking for. Therefore, this camera is no longer under consideration.

3.5.1.3 ArduCAM USB Camera Shield

The ArduCAM USB Camera Shield is a universal camera control board designed for PCS and embedded systems like the Raspberry Pi. The ArduCAM USB Camera Shield will be used in synchronization with either the OV2640 2 MP lens or OV5642 5 MP lens. It comes with software that allows you to work on Windows and Linus systems with a complete SDK library. One of the ArduCAM USB Camera Shield's perks is that the SDK it comes with is fully integrated with Python. This allows our camera to support OpenCV based applications. The ArduCAM integration with OpenCV made it a top consideration for which camera was chosen in the end. Since it is an integration that allows OpenCV based applications it does not really restrict us on how we would like to implement our CV application in the garage. Unlike the Pixy Smart Vision Sensor, which restricts us with the use of the hue-based color filtering. There would be some setbacks if we were to use the ArduCAM. The main issue is that it is built specifically for PCs and embedded systems which the Arduino is not. Another limitation with the ArduCAM is its bandwidth. The camera shield has no onboard frame buffer which is used for the representation of the content to be shown. The transfer reliability depends on the USB bandwidth, so if there are more than one USB devices, it will cause drop frames. This can really hurt the accuracy of our CV results. This doesn't mean we should disregard the ArduCAM because it also has some really great features.

Some other great qualities about the ArduCAM USB Camera Shield is its USB control interface, and the ability to choose the camera module. Since this camera has the plug and play USB control interface it makes it incredibly easy to use with computer hardware. When it comes to the software it is also simple because of the SDK library that comes with it. The ArduCAM is said to work without having to do any modifications from the start. In our case we would want to add much more than just the standard package. The ArduCAM should be able to support anything within our range of needs. From creating our own CV applications, to returning any data necessary to return to our base unit. The USB control interface gives it a great look because of our idea of wanting to keep the system simple. Being able to choose the camera module is also a great quality because it gives us a chance to adjust our accuracy for our CV results. This is somewhere we should not keep it simple because of accuracies importance in this project.

3.5.1.3.1 OV2640 2 MP

The first camera module that was put into consideration was the OV2640 2 MP. This camera module with the camera shield can be bought and used at the time being for $34.99 each. Some features that come with this camera module are High sensitivity for low-light operation. This would be useful in case we were dealing with a garage with bad lighting or in the case that there is a light that is broken in the garage. It also comes with image quality controls that include color saturation, sharpness (edge enhancement), lens correction, white pixel canceling, and noise canceling. This can all also be used for the benefit of acquiring accurate information from our CV application. There are three different specifications for power supply. Core is 1.3 V DC +- 5%, analog 2.5 ~ 3.0 V DC, and I/O 1.7 V to 3.3 V. The power requirements are separated by active and standby. For active they are 125 mW (for 15 fps, UXGA YUV mode), and for standby its 900 uA. It has a lens size of ¼" and a chief ray angle of 25 degrees non-linear. The maximum image transfer rate is dependent on the resolution type. For UXGA/SXGA its 15 fps, for SVGA its 30 fps, and for CIF its 60 fps.

3.5.1.3.2 OV5642 5 MP

The only other camera module that was considered was the OV5642 5 MP. This camera module with the camera shield can be bought at used at the time being for $47.99 each. The power supply is separated by core which is 1.5 V DC + 5 % (internal regulator), analog 2.6 ~ 3.0 V, and I/O 1.7 ~ 3.0 V. Its maximum image transfer rate is dependent on the resolution. For 1080P (1920 x 1080) its 30 fps, for 720p (1280 x 720) its 60 fps, for VGA (640 x 480) its 60 fps, and for QVGA (320 x 240) its 120 fps. Its lens size is ¼" and has an input clock frequency of 6 ~ 27 MHz. This camera module is no longer in consideration because of its price compared to the OV2640 2 MP. The OV2640 2 MP should be able to do anything necessary for our project, so there is no need to pay an extra $13.00 per device.

3.5.1.4 OV7670 with FIFO

The OV7670 camera will be considered only with the FIFO memory included because without it comes with difficulties of trying to interface because of the pixel clock. This camera comes with a few perks which include its very low price and its small size. It is also low power, so this camera really focuses on doing simple task. It can be useful in keeping our project really simple and cheap, but it could also jeopardize our project by not bringing enough accuracy on the CV part. The team holds a high priority on keeping the project low cost and simple which is why the OV7670 was a top consideration. The camera alone can be acquired for under $3.00 but that would result in a lot of extra unnecessary work. That is why we included it with the FIFO memory because without it we are very limited by the memory. Some major downfalls with this camera is that the documentation for the camera is not completely available. This is a major problem in synchronization with the problem that it is not that simple to set up with the Arduino. This camera is also very slow at capturing images and the only reason it stays as a consideration is because our project does not care so much about dynamic objects as it cares about static objects.

The OV7670 with FIFO can be bought and used at the time being for around $9.99 each. The dimensions of the device are 30.5 x 30.5 mm. It also comes with high sensitivity for low-light operation the benefit of this as described earlier is when applying our CV application, we can still use it if there is a light that is not working in the garage. It has low operating voltage for embedded portable apps. This is useful in keeping our project low cost. Has Exposure Control (AEC), Automatic Gan Control (AGC), Automatic White Balance (AWB), Automatic. As with some of the other cameras it also includes image quality controls like color saturation, hue, sharpness (edge enhancement), and anti-blooming. This is all useful for receiving accurate information from our camera to be interpreted by our CV application. There is also an edge enhancement level auto adjust. For the power supply only a single 3.3 v supply is needed.

3.5.1.5 Arducam 5 MP OV5647

The Arducam 5 MP OV5647 was created to meet the increase need for Raspberry Pi compatible camera modules. This camera module will not be used with the ArduCAM USB Camera Shield because it is not compatible. This may seem like a downfall, but it results in some great qualities because it removes the downfalls of the ArduCAM USB Camera Shield. The Arducam 5 MP OV5647 does come with one major setback, but only if we were to use the Arduino as our base unit. Seeing as this camera is meant to be used with more specifically the Raspberry Pi. In the other hand if we were to use the Raspberry Pi it would eliminate this issue.

The Arducam 5 MP OV5647 comes with the freedom to implement OpenCV in any way possible. It allows us to improve our CV accuracy by allowing us to create our own CV methods as well as testing until a satisfying result is found. Using this camera would also be simple if used with the Raspberry Pi. This interface uses the dedicated CSI interface which was designed just for these two components. The CSI bus is capable of high data rates, and it exclusively carries pixel data. This completely avoids the issues of transfer reliability that depends on USB bandwidth, which would drop frames. It also allows for a much better picture quality and the ability to interrupt data much faster and accurately. The Arducam 5 MP OV5647 can be bought and used at the time being for $14.99 each. This very low price compared to some of the other cameras has made this a top contender. The sensor has a native resolution of 5 megapixel, so there is no doubt it will have enough quality to detect cars with a human eye. It also has a fixed focus lens onboard and supports 1080p30, 720p60, and 640x480p60/90 video. It is also a small unit of 36 x 36 mm dimensions. Another great aspect of freedom with this camera is that the lens is replaceable.

3.5.2 Camera Comparisons

Initially there were a bunch of cameras being investigated, then there were six that seemed to have potential for our project. After doing some extensive research for those six cameras it was narrowed down to the top four potential choices. Cameras were eliminated based on price, compatibility with our project, and the ability to use CV with accuracy. Some cameras were doing extra things that we did not

really need, which resulted in an up in price. Some would not be able to give accurate information for out CV application.

3.5.2.1 Cost

The system cost is one of the most important aspects to our project. Since smart garages is such a popular project we want to renovate the idea by making it low cost and more accurate. Also, having an affordable system can determine whether client will take up the system or remove it from their consideration

Table 9: Comparison of camera cost.

Camera	Unit Price ($)	Total Price (x2)	System Price Increase (%)
OV7670 with FIFO	$9.99	$19.98	0%
ArduCAM USB Camera Shield with OV2640 2 MP	$34.99	$69.98	250.25%
Pixy Smart Vision Sensor	$69.00	$138.00	590.69%
Arducam 5 MP OV5647	$14.99	$29.98	50%

The table above shows that the OV7670 with FIFO is by far the cheapest option based on percentage. Even though cost is an extremely crucial factor to the project it is not the only one. The table here only looks at the cost of the unit itself, not all the necessary parts that would be needed to make them compatible with the Arduino. The Pixy Smart Vision Sensor all though the highest in cost does not require any extra parts to become compatible with the Arduino. The Arducam 5 MP OV5647 even though an increase in 50% is only a difference of $5 per unit. We still need a well-rounded camera, so we should not eliminate any cameras until everything has been evaluated.

3.5.2.2 Compatibility with Arduino

The cameras compatibility with the Arduino is potentially an important aspect to our project. If our system uses the Arduino as the base unit then the camera should be able interact with the Arduino system using simple methods. This will allow the ParkShark system to stay simple by allowing compatibility of hardware and software.

The following section details the steps to set up each camera to the Arduino. The first step is to start by connecting the 10-pin cable into the camera and connect the 6-pin cable into the back of the Arduino. Then, connect the USB cable into the Arduino and connect the Mini USB cable into the back of Pixy Cam. Next, install the latest update at [1], get the latest version of PixyMon and Arduino libraries. After downloading the latest versions, open the Arduino and go to the sketch tab,

import library, then add library. Choose the latest version. Go to the files tab, examples, play, select “hello_world”. Once these steps are completed, the Pixy is ready for operation. The step by step instruction on using the Pixy can be found on YouTube with a video that is 3 minutes and 53 seconds long, under the title “Pixy Cam to with Arduino”.

Next up in difficulty we have the ArduCAM USB Camera Shield with OV2640 2 MP which has some more integrate steps when compared to the Pixy Smart Vision Sensor. We start by downloading software from [2] and download the library from [3] or just clone from GitHub. Then you want to unzip the Arduino-master.zip file that was just downloaded. You want to select the ArduCAM, ArduCAM_Touch, and UTFT4ArduCAM_SPI libraries folders. Then copy them to the directory under \Arduino\libraries. Then open the memorysaver.h under the same directory as before Arduino\libraries\ArduCAM and enable the OV2640_MINI_2MP and disable any other. Once you have this done you want to connect the camera to the Arduino board which has 8 connections to specific spots. Then configure the Arduino Ide by going to the Tools tab, Board, make sure the Arduino/Genuino UNO is selected. Then Tools tab again, Port, your port should be selected. Then File tab, Examples, ArduCAM, mini, example and then upload your code. For the final step, you will configure the ArduCAM_Host_V2 by opening the path to the ArduCAM_Host_V2.0_Windoes and pick the right Port and Pix. Once you have that click open and capture to get started. This set up also has a step by step instruction video on YouTube which is 6 minutes and 24 seconds long, under the title “Arducam 2MP mini camera for Arduino tutorial 2018”.

Next, we have the OV7670 with FIFO which is by far the most complicated setup by a large margin. This set up has some very specific requirements that must be meet otherwise there is no way to set it up, unless there is an alternative which has not been found. The first requirement starts with the hardware. You will need a voltage level translator to connect the 3.3 V sensor to the 5 V microcontroller. It is not certain if this set up will work with any other Arduino besides the Arduino Mega. The module being used must have 18 pins with specific setups. If it does not have the appropriate pins it won’t work. You will also need jumper wires and breadboard to make the connections. Then you will make 15 specific connections to the board. Once you have this you can get the necessary software needed to communicate with the Arduino via USB at [4]. According to the article there are some issues with this and you will need to adjust them accordingly. Once that is done you will change the serial port name to match your stems name. If you manage to get this far I believe you are ready to get started. For this step by step instruction there is not a video instead there was an article explaining a project setup. There are so many uncertainties when it comes to setting up the OV7670 with FIFO that it made the team reconsider it as a possibility. We did not want to disregard it until all factors were made so we still had it as an option, but it was not very likely to be chosen.

Last, there is the Arducam 5 MP OV5647 which unfortunately has little to no known information about its compatibility with the Arduino because it was made for the sole purpose of being used with the Raspberry Pi.

3.5.2.3 Compatibility with Raspberry Pi

With the other main contender for the base unit being the Raspberry Pi, the group felt the need to compare how compatible each one of these potential cameras are with this unit. As stated before, the group would prefer a system that would remain simple by maintaining as much compatibility as possible between the various components of the device.

The following are steps for setting up the cameras to connect to the Raspberry Pi. The first steps will be simple while later steps will be more difficult. The simplest step is connecting the Arducam 5 MP OV5647. The first thing to be done do is unplug the Raspberry Pi and then connect the ribbon cable to the camera connector on the Raspberry Pi. Next, boot up the Raspberry Pi and run these two commands sudo apt-get update, and sudo apt-get upgrade. This will give the latest version of the available software. Once done run the command sudo raspi-config and enable the camera module. Then reboot the system and you are ready to begin. The step by step instruction can be found on YouTube with a video that is 4 minutes and 58 seconds long, under the title “Raspberry Pi Camera Setup Tutorial for Beginers”.

Last, is the Pixy Smart Vision Sensor which does not have as simple setup as the Arducam 5 MP OV5647. This is the only other camera that is being considered with the use of the Raspberry Pi because after some further research it was decided if that if the other cameras were to be used, they would be used specifically with the Arduino. The first step to setting up the Pixy to a Raspberry Pi is to connect the Raspberry Pi to the Internet using either an ethernet cable or Wi-Fi. Next you want to connect the Raspberry Pi to a video display using an HDMI cable. Once you have that done you connect the keyboard, mouse, and the Pixy cam to the USB ports found on the Raspberry Pi. Insert the NOOBS SD Card and connect power to the Raspberry Pi. Then you will have multiple software dependency installations to complete. This is dependent on what you are trying to do. Unfortunately there was no video for this setup instead there is a website [5] which gives steps to trying to set up the two devices. After further research there were issues that we discovered with using the Pixy with Raspberry Pi. Many sources explained that this setup could require jumper cables because the connection through USB would not properly setup the two devices.

3.5.2.4 CV Accuracy

The system’s ability to accurately determine if a parking spot is available is the whole idea behind our project, so having our camera implement CV accurately is one of the most crucial factors. Without accurate information our project is destined to have too many issues. The system cannot afford to lack in this area. It is important to choose a camera that can properly complete this task. This will all be determined based on the cameras specifications. More specifically where the cameras lack. For example, the ArduCAM USB Camera Shield with OV2640 2 MP has a major issue with its transfer reliability. It is dependent on the USB bandwidth which can be slowed down by the CPU loading changes. Another issue is it not having a frame buffer, this could either make our camera lag or chop our images.

It comes with a (1600 x 1200) resolution, 25-degree non-linear chief ray angle, and holds a stable image for temperatures ranging from 32 degrees Fahrenheit to 122 degrees Fahrenheit. The OV7670 with FIFO comes with its own problems as well. The camera itself being so cheap comes with some big setbacks when it comes to specifications. It only has a resolution of 640 x 480 VGA which leads to a less accurate representation of an object. This camera also has a 25-degree field-of-view and holds a stable image for temperatures ranging from 32 degrees Fahrenheit to 122 degrees Fahrenheit. Next, we have the Pixy Smart Vision Sensor which already comes with CV implemented with the camera which is an incredible perk. It has a great field-of-view of 75 degrees horizontal, and 47 degrees vertical. The Pixy Cams sensor has a resolution of 1280 x 800 but because of its RAM limitations it does not process a frame that large, so it results in a resolution of 320 x 200. There is currently no information on the temperature range for the Pixy Smart Vison Sensor. Last, we have the Arducam 5 MP OV5647 which is a high-definition video camera. It also has a 5 MP sensor and still picture resolution of 2592 x 1944. The maximum video resolution is 1080p and maximum frame rate of 30 fps. This camera has a replaceable lens which can be replaced with lens with 75 degree to 160 degree field of view.

3.5.3 Camera Selection

The OV7670 with FIFO had a very strong start when comparing the four cameras up for final consideration. It's very low cost makes it very appealing at first but once you go deeper into it you find that the price comes at a big cost. You would have to spend on average an extra $26.00 to make the device compatible with an Arduino system. This kind of defeats the purpose of the camera and just takes away the one major advantage it had over the other two considerations. With the camera also being so cheap it comes with some very low specs. To make matters worse it is not a guaranteed that the camera would work properly with the Arduino setup if chosen. Due to all these factors, the OV7670 with FIFO is no longer a consideration for our system.

This leaves us with three cameras to choose from, the ArduCAM USB Camera Shield with OV2640 2 MP, the Pixy Smart Vision Sensor and the Arducam 5 MP OV5647. These cameras are reasonable for our parking garage project. The OV2640 out performs the Pixy Cam in price by almost double, but the Pixy Cam out performs the OV2640 when it comes to the Arduino compatibility. On the other hand the Arducam 5 MP OV5647 is much cheaper than either of those two options by a large margin by percentage and unit price. The Pixy Cam was made to be easy when dealing with an Arduino. The OV2640 is manageable but it takes away from our idea of keeping our project simple. If using the Raspberry Pi then the Arducam 5 MP OV5647 would be our only option seeing as how the Pixy Cam lacks its ability to properly work with it. The three remaining cameras can implement CV accurately. One very important factor that the Pixy Cam has over the OV2640 is that it already comes with CV implemented from the start. This holds with our idea of keeping it simple. As for the OV2640 we would have to find a way to set up our OpenCV while using an Arduino. As explained earlier OpenCV is not portable to microcontroller systems so we would have to figure out a way to

implement CV applications with the OV2640, while keeping it simple. For this reason, the OV2640 is no longer a consideration for our system. Lastly, the Pixy Cam not allowing the use of alternative methods besides its hue-based color filtering algorithm to detect objects made it less attractive. As it limited our freedom on how much improvements we could make with our CV methods. Also, the Pixy Cam's price is incredibly difficult to justify as it would greatly increase the price of our system. For a garage with multiple floors this project would soon become too expensive.

After weighing our pros and cons for each camera, the team decided to use the **Arducam 5 MP OV5647 with the Raspberry Pi** for the Park Shark system. It lets our project stay simple in areas that can lead to many problems but leaves plenty of freedom in important areas like the CV side of the project. It allows our system to cover more area while staying at a competitive price.

3.6 Wireless Communication Considerations

For this project, the camera modules and the parking spot sensor modules require a way to communicate with each other and communicate back to the server running the parking status backend software. Ideally, a long range wireless communication technology would be necessary to create a network connection consisting of all the sensors in the sensor system. The sensor system would also require a communication technology to transmit their status to the camera system that's connected to the server and in charge of updating the parking status. The computer vision system would require internet access to send the data collected to the server and update the statuses of the parking spaces. In this section, various wireless communication technologies were taken into consideration and compared to one another.

3.6.1 Wi-Fi

Wi-Fi, or wireless networking, is a method of wireless communication that uses radio waves to transmit signals between devices. Wi-Fi is used to provide Internet access to devices. Wi-Fi transmits signals in frequencies of 2.4 GHz, 3.6 GHz, or 5 GHz. The frequency used depends on the amount of data being sent across the network.

There are two types of Wi-Fi modes: infrastructure mode and ad-hoc mode. In infrastructure mode, a router or central access point is necessary for devices to connect to and provide Internet access. In ad-hoc mode, a router isn't required. An ad-hoc network is made for temporary usage meanwhile an infrastructure is for permanent usage, so it won't be considered for this project.

A wireless network generally consists of a wireless adapter, a wireless router, and a wireless access point. A wireless adapter is used to translate data into a radio signal. This signal is transmitted using an antenna to a decoder known as a wireless router. The decoded data is sent to the Internet using a wired Ethernet connection. The router can also transmit information collected from the Internet to the wireless adapter. A wireless router creates a wireless network consisting of all

the devices connected to it. Wireless routers that run on a 2.4 GHz band have an operation range of 90 meters. The range can be slightly affected by some materials walls are made of.

3.6.1.1 Advantages and Disadvantages

Wi-Fi Advantages

1. Wi-Fi routers operating on 2.4 GHz have a distance range of up to 46 m indoors and 92 m outdoors.
2. Wi-Fi offers high speeds and is capable of, for example, 1300 Mbps (standard 802.11ac).

Wi-Fi Disadvantages

1. Wi-Fi has poor security in which anyone who is within range could potentially use it or hack it, especially if not secured with a password.
2. Interference can occur due to walls, other electronic equipment, and the distance between the device and the wireless router.
3. Wi-Fi is a bit more expensive compared to Bluetooth technology.

3.6.1.2 Decision: The system will use Wi-Fi

For this project, the Park Shark Team decided to implement **Wi-Fi communication** due to the following reasons. First of all, Wi-Fi allows us to connect the computer vision system to the internet. This connection will allow us to update the status of the parking spaces on the website we will create. Also, Wi-Fi offers an ample amount of operating range between the device and the wireless router. This allows us to place the wireless router where we find most convenient as long as it's placed in the specified range. As a bonus, Wi-Fi also offers a high speed of 1300 Mbps.

For the camera system, Wi-Fi communication will be used to establish a connection between the camera system and the Internet. One of the camera modules in the computer vision system will act as the link between the entire system and the Internet. It will be in charge of sending information relayed by the sensor system and the computer vision system over the Internet to update the status of each parking space on the website and the mobile application. To establish a Wi-Fi connection, only a wireless router would be necessary since the Raspberry Pi has a built-in wireless adapter.

3.6.2 Infrared

Another common method of communication between devices to be considered is IR, or infrared, communication. IR light is invisible to the human eye because it has a longer wavelength than visible light. The frequency range for infrared spans from 430 THz to 300 GHz (700 nm to 1 mm in wavelength) in the electromagnetic spectrum. IR communication has a range of 10 to 30 meters.

There exist two link types in infrared communication: point to point and diffuse point. Point-to-point requires the transmitter and receiver to be pointed directly at

each other with no obstacles standing in the way of their line of vision. For example, a remote control utilizes point-to-point IR communication to control the television. Unlike point-to-point systems, diffuse point systems aren't affected by obstacles and don't require the transmitter and receiver to be pointed at one another. Instead, the signal sent by the transmitter reaches the receiver by reflecting off surfaces such as the ceiling as long as they are located in the same vicinity. Both linking types are displayed below in Figure 14. For this project, the diffuse system would be the ideal linking type because there are various obstacles present in a parking garage.

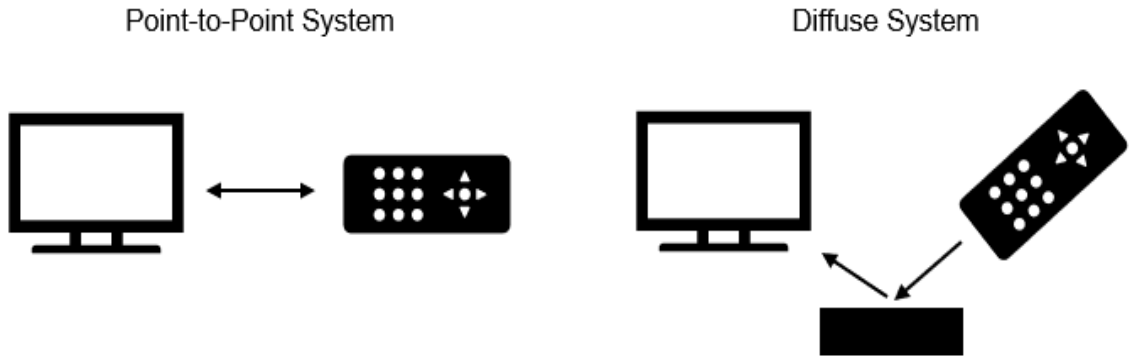


Figure 14: Linking types of IR communication.

Infrared communication involves two separate components: a transmitter and a receiver. One sends a signal while the other receives it. The transmitter converts an electrical signal to an optical signal and sends the signal to the receiver by emitting an infrared light at a certain pulse. Some devices used as transmitters are light-emitting diodes (LED) and semiconductor laser diodes (LD). Since various IR sources exist in the environment, modulation must be implemented to differentiate the transmitted IR signal from the other IR sources and to avoid interference. Modulating a signal is essentially giving the data a unique pattern so that it can be easily picked out by the receiver. Modulation is usually implemented using a carrier frequency between 36 and 46 KHz (typically 38 KHz). Modulation will cause the transmitted signal to oscillate at the specified frequency for the time duration of the pulse. The receiver, typically a photodiode, will pick up this modulated signal, demodulate the signal, and outputs a binary waveform to be read by a microcontroller or microprocessor. The microprocessor will then be able to interpret the data sent by the Infrared transmitter and take action accordingly (such as a television changing the channel when the remote sends the signal to change it).

3.6.2.1 Advantages and Disadvantages

Infrared Advantages

1. Infrared communication offers higher security due to its need for a direct line of sight between the transmitter and receiver.
2. Infrared communication is low in cost compared to other technologies.
3. Infrared communication offers low power consumption.

Infrared Disadvantages

1. Infrared communication is meant for short range communication. Its performance drops as distance increases.
2. The transmitter and receiver require a direct line of sight (be able to see each other), so it can't go through objects.
3. Infrared communication allows only one device at a time
4. Infrared transmission is sensitive to certain environmental conditions such as dust.

3.6.2.2 Decision: Infrared will not be used

The main disadvantages to using infrared communication is that it requires a direct line of sight; it is meant for short range communication, and it can handle only one device at a time. Since this project requires a wireless technology that can connect and control multiple devices simultaneously, infrared communication will no longer be considered as an option.

3.6.3 Bluetooth

Another method for communication between all the systems involved in this project is Bluetooth communication. Bluetooth is a form of wireless communication over a short distance (up to 328 ft) using ultra high frequency radio waves. It allows one to transfer data from one device to the other. It's able to operate at 2.4 GHz ISM (Industrial Scientific and Medical) frequency band which allows it to use 79 different channels of 1 MHz each within that bandwidth. Bluetooth technology can be found in our phones, computers, headsets, etc.

This technology can connect 8 devices simultaneously as long as they're all within the same radius. This feat can be accomplished due to a technique Bluetooth implements called spread-spectrum frequency hopping. Using this technique, a device can switch to different channels within the 2.4 GHz band until if finds a channel that isn't already occupied by another device. To avoid interfering with other devices, the connected devices change frequencies about a thousand times every second. When two or more devices need to communicate with each other, a piconet is created. A piconet is an ad hoc network consisting of two or more devices enabling them to communicate with one another. In this network, a master/slave model is followed in which a single device will become the master and the others the slaves. The master is in control of the transmission in the network and the slaves follow its commands. A scatternet can also be made by forming a network consisting of two or more piconets. The general idea of a piconet and scatternet is shown in Figure 15.

Sometimes, the range for Bluetooth device depends on the environment in which it's placed. If there are physical obstacles surrounding the transmission path of the device, the Bluetooth range will be decreased. A Bluetooth device placed in an open field will be able to utilize a range of up to 100 meters. Indoors, obstacles such as concrete walls will interfere with the radio signal of the Bluetooth device, lowering the maximum effective range. Since this project deals with a system

meant for a parking garage, there might be slight interference with the concrete walls.

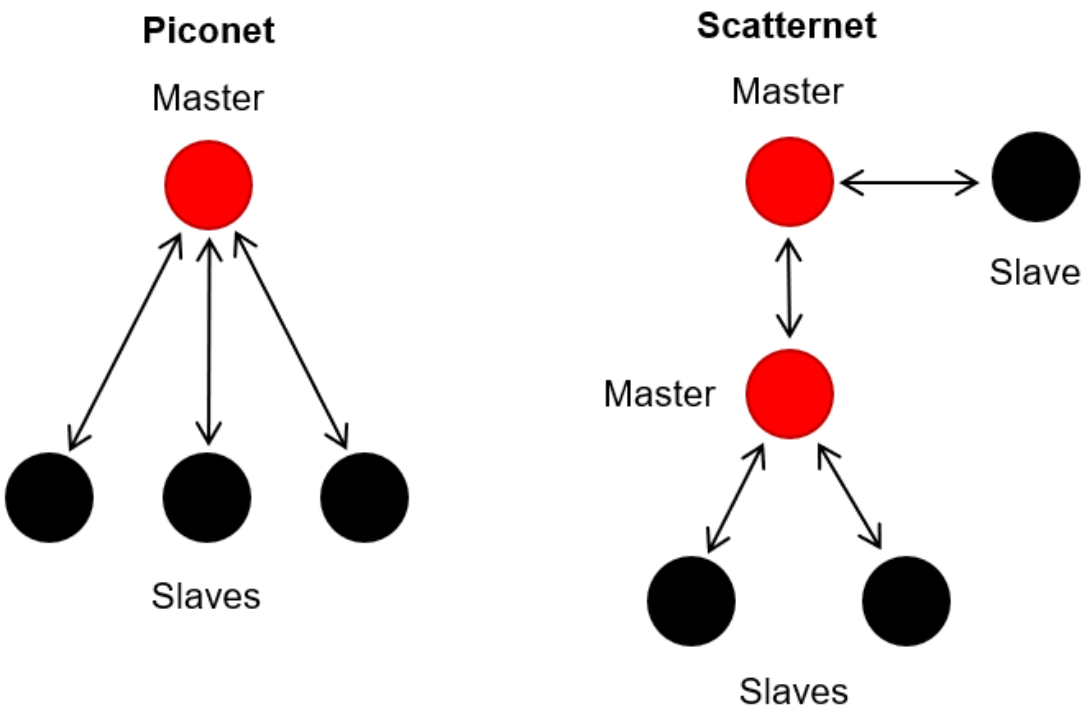


Figure 15: Visual representation of master/slave model for piconet and scatternet.

3.6.3.1 Types of Bluetooth

Classic Bluetooth was designed to handle continuous two-way data transfer for short distances. Classic Bluetooth refers to Bluetooth 4.0 and previous versions. Classic Bluetooth is typically used for streaming data, such as music or videos, due to its high data transfer rates. It utilizes 79 channels and provides a maximum of 2.1 Mbps of data throughput. In 2011, Bluetooth Low Energy (BLE) was introduced. BLE shares similarities with Classic Bluetooth such as the master/slave architecture. Also, compared to Classic Bluetooth's 79 channels, BLE utilizes 40 channels. The main difference between Classic Bluetooth and BLE is that BLE offers ultra low power consumption. For this reason, BLE devices possess the ability to extend battery life for years. Also, BLE does not support streaming due to its low data throughput of about 0.3 Mbps. While the BLE device is not in use, it remains in sleep mode until a connection is initiated once again. Instead of sending continuous data like Classic Bluetooth, BLE sends data in small packages. Hence, BLE is ideal for applications that occasionally need to transfer small bits of data. The specifications for Classic Bluetooth and Bluetooth Low Energy are displayed below in Table 10. Bluetooth Low Energy is a standard part of the Bluetooth specification as of the Bluetooth 4.0 revision, and all Bluetooth devices since then are capable of communicating with devices using the Bluetooth Low Energy standard.

Table 10: Classic Bluetooth vs. Bluetooth Low Energy.

Specifications	Classic Bluetooth	Bluetooth Low Energy
RF Frequency	2.4 GHz	2.4 GHz
Range	Up to 100 m	Up to 100 m
Data Throughput	0.7-2.1 Mbps	~0.3 Mbps
Peak Current	Up to 30 mA	Up to 15 mA
Latency in Data Transfer	~100 ms	~3 ms
Frequency Channels	79 channels	40 channels
Speed	700 Kbps	1 Mbps
Max Data Rate	1-3 Mbit/s	1 Mbit/s
Power Consumption	~0.01x to 0.5x of reference	1 (reference value)

There exist different versions of Bluetooth devices. The available Bluetooth versions are Bluetooth 1.x, Bluetooth 2.x, Bluetooth 3.x, Bluetooth 4.x, and Bluetooth 5.x. When Bluetooth 1.x first came out, it was loaded with problems and compatibility issues. Due to these reasons and its slow 1 Mbps speed, this version was barely used in actual devices. Bluetooth 2.x learned from the first generation's mistakes and became pretty popular during its time. This version introduced a new feature called Enhanced Data Rate (EDR) which increased the data rate up to 3 Mbps. It also introduced Secure Simple Pairing (SSP) which fixed the pairing issue. Bluetooth 3.x managed to improve the data rate to 24 Mbps by introducing a High-Speed (HS) feature. On the other hand, it consumes more power compared to version 2.x. In 2011, Bluetooth 4.x came out with a feature called Low Energy (LE) which enables it to collect data from LE devices. This feature also managed to reduce the overall power consumption of the Bluetooth device. Bluetooth 5.x is the latest version to hit the market offering quadruple the range and double the speed. Bluetooth 5.0 provides a data transfer speed of 2 Mbps and has a communication range of 240 meters.

3.6.3.2 Advantages and Disadvantages

Bluetooth Advantages

1. Bluetooth allows for communication through objects such as walls.
2. Bluetooth offers low power consumption.
3. Bluetooth does not require a direct line of sight like infrared communication. Data transfer is possible between two devices not in the same vicinity.
4. Bluetooth grants the ability to create a mesh network between devices.
5. Interference with other wireless networks is low due to its ability to frequency hop.

Bluetooth Disadvantages

1. There are other technologies available that offer higher transfer speeds than Bluetooth.
2. Bluetooth has weak security which hackers could potentially hack gain access to the network if in the vicinity.
3. Bluetooth is known to drain the battery. To combat this, Bluetooth low energy was developed to enhance the battery life.

3.6.4 Wireless Communication Selection

For the sensor system in this project, the idea is to have the sensors transfer their data to one main sensor that will oversee relaying that data to the camera system. If Bluetooth were utilized for the sensor system, a mesh network would be created consisting of all the sensors in the system. The main sensor will become the master and overlook all the other sensors (slaves). Since the sensors only need to indicate whether the parking spot is occupied or not, continuous data transferring will not be necessary. Also, the amount of data sent by these sensors will be a small amount. Due to these reasons, the Bluetooth Low Energy device (Bluetooth 4.x or 5.x version) would be the ideal choice. The BLE device will remain in sleep mode while the parking spot is unoccupied. Once a vehicle is detected in the spot, the BLE device will be activated and asked to relay the message that the parking spot is no longer available. By remaining in sleep mode for most of the time, the BLE device will provide ultra-low power consumption compared to the Classic Bluetooth device that's typically used to continuously transfer data. Since the Raspberry Pi used for the cameras in the computer vision system has built-in Bluetooth, Bluetooth will also be used to form a connection between the sensor system and the computer vision system. This way, we don't have to spend money on building a separate Wi-Fi module for the sensor system which would also ultimately end up consuming more power.

3.7 Bluetooth Module Considerations

The main benefit that allows the wireless sensor modules to function is that they will also communicate wirelessly. In order to utilize Bluetooth communication, a Bluetooth transceiver chip will have to be chosen to be used in the circuit. This section will compare multiple current Bluetooth 5.0 chips and select the best choice for use within the wireless sensor modules. As stated in Section 3.3.3, most modern Bluetooth transceiver chips also contain an embedded microcontroller. Due to this, it would be most economical and most efficient to make the Bluetooth chip embedded microcontroller the central processing unit of the entire wireless parking sensor. The specifications considered in this section will also include information on the microcontrollers within each Bluetooth chip as well.

3.7.1 Texas Instruments CC2640R2F

The CC2640 is one of Texas Instruments' latest transceiver integrated circuit chips. Most notably, it features full support of the Bluetooth 5.0 spec, which has only recently been finalized. Use of Bluetooth 5.0 will give this project a competitive

edge over existing systems as it will be able to utilize the latest Bluetooth technology, which means longer communication range and lower power consumption.

Table 11: Specifications for TI CC2640R2F.

Specifications	CC2640R2F
Device Type	Wireless Microcontroller Unit
Bluetooth Standards	Bluetooth 4.2 Bluetooth 5.0 Bluetooth Low Energy
Price	$1.95 per unit
Microcontroller	ARM Cortex M3 16-bit architecture 48 MHz Clock Speed 273KB NVRAM
Voltage Input	1.8 V to 3.8 V

One disadvantage of the CC2640R2F or any Texas Instruments Bluetooth chip is that Texas Instruments has not yet implemented the Bluetooth Mesh standard, and they have not implemented it as an alpha-test either, meaning that such support will not likely arrive until months or perhaps even years from the project's design date.

3.7.2 Nordic nRF52832

Another large-scale Bluetooth chip manufacturer is Nordic Semiconductors. Nordic has also recently released a new line of Bluetooth 5.0 chips. Nordic is also well-known for offering support for small scale developers rather than only large quantity buyers, meaning that if the project design group was having serious trouble getting a part of the Bluetooth chip to work, the group could likely contact Nordic for assistance and receive advice and mentoring on the proper design. Otherwise, there is also extensive online information for the Nordic nRD52 platform, including reference designs.

The NRF52832 was not initially a Bluetooth 5.0 chip, but a Bluetooth 4.2 one. However, Nordic was able to retroactively add some features of Bluetooth 5.0 to the nRF52832, making it somewhat compliant to the Bluetooth standard. The main Bluetooth 5.0 feature that the nRF52832 does not provide is the Bluetooth Long Range support. However, Nordic does have an early implementation of the Bluetooth Mesh standard on all of their recent nRF52 chips, meaning that even without the attractive feature of extended wireless communication range, the disadvantage could be mitigated by making use of Bluetooth Mesh. This makes the nRD52832 a viable option for use within the wireless parking sensor.

Table 12: Specifications for Nordic nRF52832.

Specifications	nRF52832
Bluetooth Standards	Bluetooth 4.2 Some Bluetooth 5.0 Bluetooth Low Energy Bluetooth Mesh
Price	$3.51 per unit
Microcontroller	ARM Cortex M4 32-bit architecture Fixed Point processing 512KB NVRAM
Voltage Input	1.7 V – 3.6 V

3.7.3 Nordic nRF52840

The NRF52840 is Nordic's most recent Bluetooth chip. It fully integrates the Bluetooth 5.0 feature set, and also supports Nordic's alpha-testing implementation of the Bluetooth Mesh standard. The chip also supports a higher range of voltage input, which could reduce the cost of the overall circuit by potentially removing the need for an external voltage regulation module. However, there are several large drawbacks for the nRF52840. One drawback is its high price. Furthermore, the group found that the nRF52840 is not in stock on Digikey or Mouser, meaning that procuring chips to use in the project will be difficult and unreliable, as the group would have to wait and hope for new batches of the nRF52840 to arrive promptly. The reason for the supply shortages is likely that the nRF52840 is a very new chip and has not ramped up to the production speed and demand of the earlier nRF52832.

Table 13: Specifications for Nordic nRF52840.

Specifications	nRF52840
Bluetooth Standards	Bluetooth 5.0 Bluetooth Low Energy Bluetooth Mesh
Price	$7.08 per unit
Microcontroller	ARM Cortex M4 32-bit architecture Fixed Point processing 512KB NVRAM
Voltage Input	1.7 V – 5.5 V

3.7.4 Bluetooth Module Selection: nRF52832

Initially, the group decided that the nRF52840 was the most attractive option, due to its integration of the increased range feature of the Bluetooth 5.0 standard. However; the nRF52840 is not currently available on Mouser or Digikey, with an estimated batch to arrive mid-summer. Unfortunately, this availability proves to be too unreliable to design the project around, so the group has decided to use the readily available **nRF52832** instead. However, because the chips are otherwise very similar, including the pins and feature set, the group may consider adapting the design to use the nRF52840 instead if a reliable supply for the chips happens to be found over the summer. This would allow the wireless sensor module placed at individual parking spots to have a much higher connection range, greatly enhancing the usefulness of the sensors and reducing the complexity of placing the master unit (the computer vision camera module) close enough to the sensors for communication. Either way, both chips still support an alpha-version implementation of the Bluetooth Mesh standard, which will also assist in laying out the networks of parking spot sensors. The TI CC2640RF was discarded from consideration as Texas Instruments has not implemented Bluetooth Mesh support into any of their Bluetooth transceiver product offerings.

Table 14: Comparison of Bluetooth transceivers.

Feature	CC2640R2F	nRF52832	nRF52840
Voltage Input	1.8 V to 3.8 V	1.7 V – 3.6 V	1.7 V – 5.5 V
Processor	ARM Cortex M3	ARM Cortex M4	ARM Cortex M4
Bluetooth 5.0 High Range	Yes	No	Yes
Bluetooth 5.0 Reduced Power Consumption	Yes	Yes	Yes
Bluetooth Mesh	No	Yes	Yes
In Stock on Digikey or Mouser	Yes	Yes	No
Price	$1.95	$3.51	$7.08

3.8 Mobile Application Technologies

The Park Shark team will be developing a mobile application alongside a web application. The mobile application is supposed to tell users the availability for the garage that are monitored by the Park Shark system. The mobile application is important because for a modern-day technology to be relevant and able to

compete with other existing technologies it should come with a mobile application. The majority of college students use their cell phones on a regular basis. Without the mobile application our Park Shark system would not appeal to our audience seeing as it would be used most commonly by college students, more specifically the University of Central Florida's students. The mobile application will be receiving information from our base unit.

3.8.1 Mobile App Technology

Mobile applications can be separated into three major branches: web applications, native applications, and hybrid applications. For the mobile application we are considering between a native application or a hybrid application. The web application is not considered for this portion because it has its own focus in the web development portion of this paper. Our goal is to make an application that students can download from either the Google Play store for Android users or the App store for iPhone users.

3.8.1.1 Native Applications

Native applications are the most common type of mobile applications. They are built for specific platforms. If you are working on Android, you are likely to be using Java. As for the iPhone you would most likely be using Swift or Objective C. Native applications are known to be superior in performance because they are built to run on a specific platform. Native applications make it easy to work with device utilities like the camera, speaker, and GPS. The fact that it runs on a single platform can make it difficult to maintain and will result in more work. It will require the knowledge for each platform.

3.8.1.2 Hybrid Applications

Hybrid applications are a combination of native and web applications. It has the advantage of using the easier to learn and use languages like HTML, CSS, and JavaScript which is what is used for the web applications. It has the advantage over a web application because it can be used like a native application in the sense that it can be put on the Google Play store and App store. It acts like a native application and it does this by running inside of a container. One major advantage with the hybrid application is that it will work with all platforms unlike the native application. Like the native application it can use the device utilities by using an API. The hybrid applications dealing with the wrapper frames make it slower than the native applications. It also comes with some difficulties when dealing with UI.

3.8.2 Mobile Application Choice: Hybrid App

The determining factor for choosing to work with a hybrid application is its simplicity. Instead of having to create separate platforms for different devices like iPhone, Android, BlackBerry, and Windows. We would only be creating one platform that can be used by any device. We want to make sure the users experience with the mobile application is satisfied. Most users would not give an application a second or third chance if they had a bad experience with it. Therefore, we want to ensure we get it right the first time. The Park Shark team believes that

creating a hybrid application even though slower than the native applications can be done with enough user satisfaction.

3.8.3 Mobile Framework Options

There is an abundance of mobile frameworks to work with nowadays. With so many options it can be difficult to narrow down the best options to consider. Since the Park Shark team has little knowledge on mobile app development we decided to choose from frameworks that were popular. This would give us the resources necessary to get started on developing a mobile application with ease. After doing some research on the more common mobile frameworks available we decided to do more in depth research on React Native, Ionic, Xamarin, and PhoneGap.

3.8.3.1 React Native

React Native allows its user to build mobile applications by using JavaScript and React. JavaScript is the only programming language you will need to work with this mobile framework. A great advantage to react native is that the app will be indistinguishable from apps that are build using Objective-C or Java. React Native uses the same user interface ingredients as iOS and Android applications. The only difference is that React Native does this by using JavaScript and React. When working in React Native you will be writing small pieces of code that describe how a portion of your app should look. You will determine this based on some input data. These pieces of code are the building blocks to create a full on mobile application. They are easy to reuse and can be used to construct components that represent a scene.

When dealing with hybrid applications you are dealing with a framework built on top of the mobile application. This means that the mobile application would not match the feeling and look of a real native application. Things like navigation, scrolling ease, and keyboard behavior would not be the same which can lead to frustrated users. With React Native even though you are working with JavaScript the components you make will end up translated as native platform widgets. One great advantage here is that we can use React to create our web application which would keep our software development in parallel. Although they are not the exact same it would make for an easier transition. Many of React Native's components are written in Java or Objective-C. The Park Shark team all have some experience with Java and some with Objective-C, so this is not a problem. Reactive Native also gives an advantage to the developer when building UIs. UI bugs are a lot easier to track because you define what your view should be based on input data and React will handle the rest. Another advantage to the developer is the development environment used. The environment allows you to make changes to your code without having to recompile and restart your application every time you want to test a new change.

3.8.3.2 Ionic

The Ionic framework uses HTML5, CSS, and JavaScript to write and run applications. It also requires the use of Cordova wrapper. The Cordova wrapper allows access to native platform controllers like the camera and GPS on the user's

phone. One major advantage of using Ionic is its cross-platform portability, it allows for up to 98 percent of reusable code. Another advantage is its ability to quickly test because it can be done instantly on a browser. Ionic comes with some major downfalls because of its use of web technology to render an application. This will reduce the speed of our application alongside it not being meant to be used for complex applications. The Ionic framework is no longer in consideration because purpose of being used for simple or corporate applications. The design and complexity of the mobile application is still in question resulting in a major issue if we wanted to use Ionic.

3.8.3.3 Xamarin

The Xamarin framework is supported by Microsoft and is used for mobile application development by using C# and native libraries wrapped in the .NET layer. The are many great advantages of using Xamarin over other cross-platform frameworks. Xamarin allows for up to 96 percent of the source code to be reused. It also does not require switching between development environments, they can build the application by using Visual Studio. Some of the Park Shark team members have experience with Visual Studio and C# so this would not be a whole new learning experience. Unlike tradition hybrid applications, Xamarin can create a mobile application that come close to native applications performance. Just like having native applications performance it also comes with a similar user experience. Xamarin comes with platform specific UI elements which converts application UI components into the platform-specific interface elements at runtime. This is done by using Xamrin.Forms tools which increases the speed of development but does take a slight toll in performance because of the extra abstraction layer. Xamarin avoids issues with native platform controllers by using plugins and specific APIs, to work with the platforms functionality. Xamarin has the option of using Xamarin.iOS and Xamarin.Android for more custom UIs. If code sharing is a higher priority, then you can just use Xamarin.Forms. Xamarin.Forms allows for rapid prototyping. Microsoft supports the Xamaring community by creating many resources available to the users. Xamarin University is a great example because its whole purpose is to help people who are just getting started with Xamarin. It helps with experienced users with C# as well as users with little programming experience.

The Xamarin framework still comes with some drawbacks. One obvious reason is with the community it holds. This can lead to a lack of resources when running into issues. Another issue comes when dealing with Xamarin.Android and Xamarin.iOS because you will need to know some basics for the platform-specific layer. This would require you to know Java for Xamarin.Android which is not a problem for the Park Shark team. As for Xamarin.iOS you would need to know Swift and Objective-C which is not something the team is comfortable with.

3.8.3.4 PhoneGap

PhoneGap allows its user to make hybrid applications by using HTML, CSS, and JavaScript. Like the other frameworks being compared PhoneGap allows for multiple platforms with a single codebase to reach all types of devices. PhoneGap

is a cloud-based service, which allows for the use of the most recent SDK for the targeted platform. One great advantage to using PhoneGap is that it is open source. It provides the advantage of having the developer community along your side. You will also have access to the PhoneGap toolset. PhoneGap also provides a mobile app to connect to the development machine, then see changes you are making in real time. Which is an amazing simple way to test your product. PhoneGap's official websites also comes with plenty of resources from tutorials to documentation. Another great advantage to PhoneGap is its accessibility to Native APIs so it can help retain the native application experience. PhoneGap also comes with some setbacks. First of is that Apache Cordova which is like the engine that runs PhoneGap is known to not be complete. There can be issues of wanting to find a plugin and finding it may not be the issues as open source really helps here, but that it may be out of date or not supported on the targeted platform. As with other frameworks that use Cordova the performance may lack. If there are too many graphic elements you are almost certain to run into some issues. There are ways of trying to fix these issues like using 3rd party software.

3.8.4 Mobile Framework Comparisons

Initially we were faced with an abundance of mobile application options. Then after doing some further research we decided on developing a hybrid application because of less constraints, but once again we were faced with still too many options. Once we had an idea of some potential frameworks we were left with four options. Once we made extensive research on those potential frameworks it was narrowed down to the top three potential mobile frameworks. The framework that did not make the cut was eliminated do to its purpose of being used for simple or corporate applications. The remaining mobile frameworks will be compared base on community size and learning curve. (Maybe "availability of resources on the market")

3.8.4.1 Community Size & Involvement

The community size is one very important aspect to a framework because it can lead to an abundance of help or a lack of help. Since we will be very reliant on our framework to help build our mobile application we want to make sure that it is going to have a large enough community by our side when we run into problems. The advantage to having an involved community is an increase in documentation on a certain technology, which would result in a reduce in time needed to learn on it. The main aspects that should be consider when looking into the community size and involvement of a frame work is how many people talk about it on the internet, how many use it, and how frequent are issues resolved.

Table 15 shows that React Native is by far the most popular in the framework community. After doing an even more extensive research on this specific topic it was discovered that PhoneGap lacks in some very well know areas. It did not have as much information on top websites like GitHub and Stack Overflow, which is a common ground for developers. On the other hand, React Native was considered the most loved framework on Stack Overflow, while Cordova was the most dreaded which is what PhoneGap uses as its engine. Right behind Cordova was

Xamarin as the most dreaded platform. All of this is under consideration when picking a mobile framework but is not the determining factor. As the community size and involvement is not the only important factoring in choosing a mobile framework.

Table 15: Comparison of mobile framework community sizes.

Mobile Frameworks	GitHub Stars	Stack Overflow Questions	Top Apps
React Native	62.9K	29.8K	Instagram, Messenger, Uber, Microsoft OneDrive
Xamarin	3.6K	30.3K	Vanderlande, Captio, CA Mobile, Novarum DX
PhoneGap	3.3K	N/A	Wikipedia, TripCase, Health Tap, Paylution

3.8.4.2 Learning Curve

The learning curve for these mobile frameworks is also very important because time can be a major constraint when developing a product. If it takes too much time to properly learn a new framework it would be major disadvantage and could possibly jeopardized our project, so we would like to ensure a mobile framework that is as easy as possible to learn. The learning curve evaluation will be based on our developers experience with languages required to use the mobile framework and the community's evaluation on learning these frameworks.

Table 16: Comparison of mobile framework learning curves.

Mobile Frameworks	Language	Learning Curve
React Native	JavaScript	Medium
Xamarin	C#	Medium
PhoneGap	HTML, CSS, and JavaScript	High

As can be seen from Table 16, PhoneGap is trailing behind the other two options. This is because even though it uses programming languages that are easier to learn, they still require someone to learn them. The group would also need to learn not just one but three different languages in order to use this mobile framework. Both React Native and Xamarin are classified as medium learning curve for their own separate reasons. First React Native is in JavaScript which would require learning a new language, which would classify it as a hard learning curve but fortunately JavaScript is consider one of the easiest languages to learn from scratch. This is why it is only classified as a medium learning curve. As for Xamarin it is being considered a medium learning curve because of criticism it has received on Stack Overflow and among many articles explaining the difficulties that come

with using it. One advantage that helps it is that it uses C# which would not require the Park Shark team to learn a new language.

3.8.5 Mobile Framework Selection

Even though the three remaining mobile frameworks were still being consider it was made very obvious that PhoneGap was lacking a bunch of key aspects once compared to the other options. PhoneGap lacked when it came to the community involvement. It did not compare to the amount of resources that React Native and Xamarin came with. Even though PhoneGap has a community that supports them it seems like they still need time to develop. PhoneGap also came with a major issue of having to learn HTML, CSS, and JavaScript which would require too much time to learn. Due to all these factors, PhoneGap is no longer a consideration for our system.

This leaves us with two mobile frameworks to choose from: React Native and Xamarin. Both of these frameworks come with some great advantages. When we look at the community of both of these frameworks they are both very well know and well involved with the community. React native has an upper hand with the community it seems though, with its 62.9K GitHub stars compared to only 3.6K. React Native also has some well-known applications that have been developed as compared to Xamarin. A key insight came when looking into Stack Overflows statistics which shows that React Native is one of the most wanted and loved frameworks. Xamarin on the other hand is on the top of the list as a dreadful framework to work with. This is incredibly difficult to ignore coming from an accredited developer cite.

After doing some more in-depth research on the remaining frameworks the Park Shark team decided to use **React Native** as our mobile framework. Even though Xamarin had an advantage of using C# it was not enough. React Native even though used in parallel with JavaScript it is well know that JavaScript is not a difficult language to learn. It also comes with a community to wants more and loves it. React Native also comes with the advantage of avoiding Cordova to improve performance. The components you create will end up rendering as native platform widgets which is as good as you can get with a hybrid application.

3.9 Website Technologies

3.9.1 JavaScript

JavaScript is a scripting language that is supported by all major web browsers used today. This language is a high level interpreted language that allows for functional and object-oriented programing design. JavaScript was initially only used in client-side browsers, but with the use of JavaScript engines such as Google’s V8 engine and runtime environments such as NodeJS, JavaScript can be compiled and run server-side. JavaScript is extremely vital in all the following technologies and will be used extensively throughout our project.

3.9.2 React

React is a JavaScript framework, developed by Facebook, that is used for building the front end of web applications. React operates off the idea of components, which is simply a module that renders HTML output. The UI that results from React will be made up from several of these components. This allows for the creation of more sophisticated UI designs as the ability to break down a large architecture into small pieces is incredibly useful especially when trying to debug or test components of the UI. These components are all independent of each other, which allows for ease of reuse and upgrading. Because of its use of components, there are only two main concepts two learn to start using React, the structure of a component and how to give those components states. Having to only learn two concepts to start building apps results in a very small learning curve and allows for ease of use and rapid prototyping. React uses a virtual Document Object Model (virtual DOM), which connects React to the actual Document Object Model (DOM) which is what the user can actually see. The virtual DOM essentially acts as a virtual browser. React updates the virtual DOM before pushing changes to the DOM, and only updates components that received updates. This results in what appears to the user as the page being rendered every change, but only rendering components that change. This allows changes to get applied in near real time. The main pros of React are the large support community, the small learning curve, and the speed that React renders output to the user.

3.9.3 AngularJS

Another JavaScript framework, Angular, developed by Google, is also used to build the front end of web applications. Angular creates the front end by extending HTML and JavaScript, adding several directives and services. Angular operates in the MVC (Model-View-Controller) architecture. The Models are objects created in JavaScript, which can then be posted to the View, which is what the user sees. The Controller connects the two by using what is called the scope, which takes the desired info from the Model and sends it to the View. Angular and its related technologies are perhaps some of the most complex compared to other JavaScript libraries, which results in a much steeper learning curve. The number of concepts required to start using Angular include modules, controllers, directives, scopes, templates, linking functions and more. For the purposes of this project, this is a major concern, as we will have limited time to learn the associated technologies as well as the desire for rapid prototyping. This is supplemented by Angular having one of, if not the biggest community of active developers supporting it, as well Google’s own documentation. Another factor to consider is the pure size of Angular. It being one of the biggest frameworks causes longer load times and performance issues when handling large amounts of data. AngularJS also adds the complexity of having its code in more than one place, with the framework being split between the JavaScript source code and the elements that are embedded in the HTML.

3.9.4 Node.js

Node.js is a JavaScript runtime environment that handles the compiling and execution of JavaScript on a server. Node is used to create the backend of web applications and handles the creation of servers and use of modules that implement various types of functionality. Node utilizes non-blocking I/O calls, which means that requests can happen in the background while Node sends data to the server, which has a large impact on runtime efficiency. Node.js has one of the biggest communities around it, making it one of the most well supported backend technologies available. NPM, which is a package manager for Node.js, has over 4 million users and allows for users to create and download packages that accomplish various functions. For example, one of the most downloaded packages on npm is the lodash library, which adds utility functions for common programming tasks such as string manipulation and math functions. This large community allows for a large amount of helpful documentation as well as the availability of libraries that can help accomplish a large amount of functionality, which allows for more time spent on the implementation of our system.

3.9.5 Express

Express is a web application framework that assists backend communication with databases. Express handles all the routes, requests, and views for the backend. All get, post, patch, delete, and other related database functionality is all handled by this framework. This makes integration with the selected backend and the selected database program much simpler, as all communication will be sent through Express as opposed to other, more complicated methods.

3.9.6 MySQL

Before we introduce MySQL, we must first describe two related technologies called Relational Database Management Systems (RDBMS) and the Structured Query Language (SQL). A RDBMS is a database structure that identifies and stores data in a way that relates with another data entry in the database. Data is stored in tables, with each column representing some form of data (such as name, or age) and have specific data types (int, string, etc.). This allows for many rows, also known as records, that can be viewed as the actual body of the database. SQL is a programming language that enables you to access and manipulate data that is stored in a RDBMS. SQL as a language reads almost as plain English, which makes the reading and writing of SQL very straight forward. SQL handles all the direct retrieve, insert, update, delete and other associated calls. MySQL is a RDBMS that uses the SQL language. MySQL, developed by Oracle, was originally released in 1995 and remains today as one of the most popular RDBMS databases to date, with companies such as Google, Facebook, and Twitter all utilizing MySQL in some capacity. The popularity of MySQL offers a large amount of documentation as well as a large support network by developers. There are also numerous packages available for backend applications such as node-mysql that makes connecting and querying MySQL via a NodeJS backend very straightforward.

3.9.7 SQLite

SQLite, developed by D. Richard Hipp and released in 2000, is another popular RDBMS. The biggest difference between MySQL and SQLite is that SQLite is embedded into the application. SQLite has no external communication with a database server; all the data is stored locally in a file-based system. This allows SQLite to be extremely fast as all data can be fetched locally as opposed to having to make calls to an external source. SQLite is best suited for embedded applications, such as systems that require portability or are single user systems. Because of this, and the requirements set for our project to create a multi-user system, the use of this database in the final product is extremely unlikely. However, SQLite can still be used in the testing process of our system. SQLite is very small, and coupled with ease of use and installation, this database can help test the final sub-systems functionality.

3.9.8 PostgreSQL

PostgreSQL, a database developed by the PostgreSQL Global Development Group and initially release in 1996, is considered one of the most advanced database management systems available today. Unlike MySQL and SQLite, PostgreSQL is an object-relational database management system (ORDBMS). A ORDBMS is very similar to relational databases but adopt an object-oriented model. In an object-oriented database, much like the principle of object-oriented programming, data is represented in the form of objects, with relations to other objects throughout the database. This differs from the RDBMS database method where all data was stored and related using tables.

PostgreSQL is completely open-source and free, compared to MySQL which offers a limited free community version as well as offering several paid alternatives. Being completely open-source has caused a very dedicated and active community that provides documentation as well as continually improving the database itself. One of the biggest upsides to PostgreSQL is the support of NoSQL features (described in detail in the next section) such as JSON, and the indexing of JSON data which allows for fast access, as well as XML support. Like MySQL, there also a large number of packages available that assist with connecting between different technologies such as node-postgres, which assists with the interfacing of a PostgreSQL database.

As mentioned previously, this database is considered one of the most advanced, and for a proof of concept project such as ours, might be overkill for such a straightforward design. A major downside to this complexity is the performance, as for simple operations such as querying, a drop-in performance compared to MySQL can be observed. What is gained from this drop-in performance is an increase in the reliability and security of the data stored. Another consideration to keep in mind, especially with the complexity of this technology, is the ability to learn in a relatively short time frame. In this department, MySQL has a leg up on PostgreSQL, as MySQL is much simpler and therefore much easier to learn. If time permits for learning and implementation, use of this database would be preferred

over others as the features and security options far outweigh any performance differences.

3.9.9 MongoDB

MongoDB is a document-oriented database program developed by MongoDB Inc and released in 2009. This database is considered a NoSQL database program and operates around the use of JSON documents. Instead of utilizing tables like in relational databases, NoSQL databases stores all data in collections which house different documents, which is simply another term for a very large JSON object. NoSQL offers a much simpler design compared to relational databases, and therefore a greater ease of use, and in very certain scenarios, faster runtime. These faster runtimes come about because of the non-structured storage of data. Instead of having to look through numerous records in a SQL database, MongoDB can retrieve a whole collection and perform one index lookup and retrieve the requested data (if all items are given a unique ID for lookup). The tradeoff compared to a relational database is the loss of structure and clean design and the loss of many helpful functions that come with a relational database. For our project, we will be storing a finite amount of data (the amount of parking spots available). Because of this, our project requires the storage of a relatively small amount of data compared to other applications. This fact coupled with the simplicity of MongoDB code, this database will be heavily considered for this project. MongoDB was also designed with the use of NodeJS in mind, with the ability to install the mongodb package via npm for immediate access and use of a MongoDB server.

3.9.10 Comparison of Website Technologies

Front-end technologies:

Of the two front-end technologies available, both are possible for use in the project. Angular offers an extremely robust and fully featured framework. It features a full MVC architecture whereas React only operates in the View of the MVC architecture. Angular has also been in open source for much longer than React, which has allowed for a very large support network and documentation to be created. Angular has several features that react does not, such as two-way databinding and services, which React requires extra software to handle these features (such as Redux). That being said, React is much easier to pick up, with only a couple of concepts needed to learn to understand the framework. React has also become one of the most popular frameworks in the past years, with the support community nearing the size of Angular rapidly. React is also a much smaller framework which results in a small increase in performance compared with Angular. Angular also splits its code between both the HTML file and the JavaScript file, whereas almost all of React will be in the JavaScript file allowing for a simple view of the application.

Table 17 shows the main criteria used to compare the two libraries. After the discussion of both technologies and the comparison table, React will be used as the front-end for web-development.

Table 17: Comparison of website frontend technologies.

Criteria	Angular	React
Size*	766K	133K
Learning Curve	High	Low
Performance	High	High
Simplicity of Code	Complex	Simple
Development Time	High	Medium

*numbers for size taken from reference [6]

Back-end technologies:

Only one back-end technology was discussed in the research section, NodeJS. NodeJS is the premier back-end for use in JavaScript. Other frameworks exist but utilize other languages such as C# and Python. Because of the preference to work with JavScript, NodeJS will be used as the back-end technology for web-development.

Databases:

Of the four database technologies discussed, only three can be used in the final product. As SQLite can only be used for testing purposes, it will not be included in the comparison table. All the discussed databases can be used to accomplish our required functionality. MySQL and PostgreSQL are very similar to each other as both are relational databases, whereas MongoDB is a non-relational document-oriented database. The main criteria used to compare the three databases came down to primarily non-technical reasons, as all of the databases have similar performances and sizes. With that said, MongoDB, in some use cases, has much faster runtimes. The reason for this difference in speed comes from how MongoDB stores data. For very large databases, the difference is negligible, but as our system will house a finite amount of data, this performance increase can be observed. Table 18 shows the main criteria used to compare the databases.

Table 18: Comparison of database backend technologies.

Criteria	MySQL	PostgreSQL	MongoDB
Learning Curve	Medium	High	Low
Simplicity of Code	Medium	High	Low
Performance*	Medium	Medium	High
Development Time	Medium	High	Low

*these classifications are for the use in our system. Outside the use in this project, all have similar performance ratings.

After the discussion above and the comparison table, MongoDB will be used as the database for our project.

3.10 Control Unit

The Computer Vision system will require a large amount of computational power to implement the detection methods. Because of this, the use of a single board computing device will be necessary. In this section we will discuss and compare three different products, the Raspberry Pi, the BeagleBone Black, and the Jetson TX1.

3.10.1 Raspberry Pi

The Raspberry Pi is a single board computer developed by the Raspberry Pi Foundation. The Pi has become a staple in the maker community, being used primarily by hobbyists and students. The reason for its popularity can be attributed to its low price and its large community involvement.

The specific model examined in this section will be the Raspberry Pi 3 model B+. The specs for this model can be found in Table 19.

Table 19: Specifications for Raspberry Pi 3.

Specifications	Values
Price	$35
Size	85.60 mm x 56.5 mm
CPU	1.4 GHz 64-bit quad-core ARM CortexA53 CPU
RAM	1 GB LPDDR2 SDRAM
Wireless Communication	Dual-band 802.11ac wireless LAN (2.4 GHz and 5 GHz) and Bluetooth 4.2
Ethernet	Gigabit Ethernet over USB. Power over Ethernet support.
USB	4 USB 2.0 Ports
GPIO	40 pins
Power	5V/2.5A DC power input
OS Support	Linux and UNIX

One of the drawbacks of the Pi is the small amount of onboard storage. To use the Pi, an external SD card is necessary to load the operating system and as its main storage. However, though this does require the purchase of an extra component,

this allows for easy installation and flashing of the system as well as transfer of data between computers.

The Pi offers an incredible amount of computational power as well as ports and GPIO pins for a very low price. The use of this device allows for fast prototyping and the use of more complex computer vision detection methods. The Pi also has a very large support network and a large amount of documentation. The Pi being one of the most popular single board computers has also created a large market for expansion shields and addons for the Pi, which could prove to be very useful for this project.

3.10.2 BeagleBone Black

The BeagleBone Black is another single board computer developed by Texas Instruments. The BeagleBone offers a great alternative to the Raspberry Pi, still offering powerful processing power as well as development options. The board is very similar to the specs of the Pi and can be used to accomplish similar tasks. The specs for the BeagleBone Black are provided in Table 20.

Table 20: Specifications for BeagleBone Black.

Specifications	Values
Price	$50
Size	86.40 mm x 53.3 mm
CPU	1 GHz AM335x ARM Cortex-A8
RAM	512MB DDR3 RAM
Storage	4GB 8-bit eMMC on-board flash storage
Wireless Communication	802.11 2.4GHz WiFi and Bluetooth
Ethernet	Gigabit Ethernet over USB
USB	2 USB 2.0 Ports
GPIO	92 pins
Power	5V/2.5A DC power input
OS Support	Linux and UNIX

The BeagleBone offers a large amount of GPIO pins compared to the Pi, which allows for use in development as well as the added benefit of supporting a large number of add-ons and shields. The Board has a decent sized community behind it, with documentation and support from Texas Instruments. The board is also very space efficient, has great on-board memory, and offers good processing speed for its size.

3.10.3 Jetson TX1

The Jetson TX1 is an extremely powerful embedded computer platform. The Jetson features a full NVIDA GPU, a 64-bit Quad ARM CPU, and a large amount of on board storage. The Jetson is used primarily for applications of advanced Robotics and AI and offers a lot of computational power to achieve these. The specs for the Jetson TX1 are provided in Table 21.

Table 21: Specifications for Jetson TX1.

Specifications	Values
Price	$499 (~$299)
Size	222.25 mm x 209.55 mm
CPU	Quad ARM A57/2 MB L2
RAM	4 GB 64-bit LPDDR4
Storage	16 GM eMMC, SDIO, SATA
Wireless Communication	802.11ac WLAN, Bluetooth
Ethernet	1 Gigabit Ethernet
USB	USB 3.0 and 2.0
Power	5.5 - 19.6V DC power input
OS Support	Linux and UNIX

For the purposes of this project, the use of the Jetson TX1 would be equivalent to the use of a supercomputer. The computational power of the Jetson would be able to run all computer vision detection methods and data transmission without much latency. The biggest drawback to this is of course the price. The board sells at retail for $499. However, NVIDIA offers a student discount of $200, which brings the price down to $299.

3.10.4 Comparison of Single Board Computers

After introducing and detailing the above boards, we will now compare each and ultimately select the board best suited for this project.

The Raspberry Pi 3 and BeagleBone black offer a very similar product, both featuring good processing power for their size, with the Pi offering a bit higher processing speed. The Pi offers a lower price compared to the BeagleBone black, but with the need to purchase an additional SD card, the end price of both becomes similar. They both have a good amount of GPIO pins and availability of add-ons, and both can be used to accomplish the requirements of this project on a technology stand point. The community support behind both boards is where the two begin to differ in a significant way. The Pi has one of, if not the biggest

communities of makers and developers using the product, as well as a large amount of well created documentation. While the BeagleBone is a very popular alternative to the Pi, the community support is not equivalent to the Beagle. For a project such as this, where the need for support will be needed throughout development, as well as the increased processing power for a similar price, the Raspberry Pi will be considered over the BeagleBone black.

The Jetson TX1 blows both the Pi and the BeagleBone out of the water when comparing hardware specifications. The Jetson offers an incredible amount of processing power as well as a dedicated GPU. All of this comes with two large drawbacks, the size and the price. The size of the Jetson Development Board is considerable bigger than both the Pi and the BeagleBone, which ultimately does not fit our size requirements for our computer vision system. The second issue with selecting the Jetson, even with the student discount, is the price. A $299 board, that while would be a fantastic choice for a project such as this, does not fit our price requirements. The possibility of a sponsorship or donation of the board would be the only way for the use of this powerful to be considered. Because of these two reasons, the Jetson TX1 will not be used for this project.

After the discussion of the three different boards, as well as the comparison between the three, the **Raspberry Pi 3 model B+** will be used as the single board computer for this project.

4.0 Realistic Design Constraints

While planning out the hardware and software designs, the Park Shark team encountered a variety of constraints that need be addressed. In this section, all the constraints that could have a possible impact on the design of the Park Shark system will be discussed.

4.1 Economic Constraints

Throughout the design period, one constraint the Park Shark team has encountered is the total cost of the project. Unlike some of the other senior design groups, the Park Shark team decided to sponsor our own project. Since our project is self-funded, the budget the team set is quite limited. Due to this constraint, the team does not have the liberty to purchase multiple items to test out. Purchasing items for testing would only increase the overall cost of the project. For example, cameras are quite expensive so purchasing a camera that doesn't meet the design criteria for the computer vision system would simply impose a financial burden on the team. Therefore, before purchasing these items, the team must conduct thorough research of each individual electronic component to ensure the best possible item was selected.

Due to the limited budget, the team will not be able to outfit the whole garage with our parking garage system. The parking garage located on the UCF campus is huge, particularly garage C which consists of 1852 parking spaces. The Park Shark team does not have the necessary funds to install a computer vision system on every floor and a sensor unit for every parking space. The budget allows the team to purchase about 2 cameras for the computer vision system and produce a few sensor modules for the sensor system. Therefore, the Park Shark team will be demonstrating the finalized product for only a small section of garage C as a proof of concept that the system works.

Also, one of the goals for the project was to create a parking garage system that won't break the bank. The cost of the parking garage system should be affordable in order to appeal to potential customers. Ideally, the team would like to purchase items of the best quality possible, but, due to the limited budget, those items cannot be purchased. To make up for the lack of funds, the team aims to select components that are low of cost, but high in quality. By doing so, the total cost of the project will be lowered without sacrificing the overall quality of the system.

4.2 Time Constraints

Another constraint the Park Shark team must take into consideration is time. The primary goal is to a create a functioning parking garage vehicle detection system by the end of Senior Design 2. The problem is the team has only two semesters to research, design, test, debug, and produce a final product. In order to achieve the end goal, the team needs to make a timeline and follow its deadlines accordingly. By following a set timeline, the team should be able to meet the primary design specifications that were set for our computer vision system and sensor system. Although, due to the time limit, it's possible that the team will not be able to create

a flawless vehicle detection system. For example, more time would be needed to completely eliminate all the kinks that might be present in the vehicle detection code written for the computer vision system. More time would also be needed to conduct enough trial and error tests to ensure that the sensor modules are able to detect any vehicle, no matter its shape or size, 100% of the time. Also, it's possible that some issues will not arise during this timeframe. Most issues arise after the system has been up and running for some time. Overall, the Park Shark team would require more than two semesters to ensure the finalized product is completely free of any faults and of higher quality.

Unfortunately, due to the time limit, some possible features were left out. The addition of these features would require more research and time to develop. For example, the addition of a solar power system to the sensor system would add more cost, take more time to research, and will complicate the design. The cons simply outweigh the benefits, especially with the time limit imposed on the team. Therefore, it was decided that these features will be considered only if the team has some time left after completing all the primary design objectives.

4.3 Environmental Constraints

Considering the Park Shark system will be placed outside in a parking garage, the system will be exposed to certain environmental conditions. The computer vision system and the sensor system need to be built to withstand a wide temperature range, humidity, and dust/dirt. A deciding factor for the electrical components making up the sensor system is the operating temperature range. Depending on the location of the parking garage, the temperature could vary from as low as freezing -69˚F to as high as 134˚F. Since the team has a limited budget and can't purchase items capable of withstanding such extreme temperatures, the Park Shark system was designed mainly with the Florida temperature range in mind. Hence, the system should be able to handle the temperature range the Orlando area tends to experience. The system should also be able to handle humidity, a common occurrence here in Florida. The functionality of the components that make up the systems should not be affected by the exposure to dust and dirt present in the parking garage.

Another issue that could be considered an environmental constraint is the layout of the parking garage. The layout of the parking garage has an impact on the placement of the entire system. For example, a major concern is the placement of the sensors modules, especially on the roof level of the garage. If the sensors were mounted onto the ceiling, then where would the sensors be placed on the top floor where no ceiling is present? The Park Shark team decided to make the placement of the sensors a bit flexible in the sense that the customer can decide where to place them. They will be given the option to place them on the ceiling, the wall, or even the floor. The same idea applies for the computer vision system.

4.4 Social Constraints

While designing the Park Shark system, some social constraints must be taken into consideration. One crucial social constraint the Park Shark team must keep in

mind is that is that people are unpredictable and so is their parking. It is assumed that people will park properly within the lines, but this is not always the case. Some people park over the lines and others take up two parking spaces for themselves. Ultimately, the way people park in parking garages has a major impact on the design's layout and software. Since people are unpredictable, placing the sensor module on the floor isn't the best option. The module could potentially get run over by a careless driver rendering it useless. If the sensor module is mounted low on the wall, the driver could hit and damage the module with their car's bumper. Therefore, the sensor's placement and housing will be designed with this constraint in mind. The computer vision coding will also be impacted by people's parking abilities. One way to detect a vehicle in a parking space is to implement a code that creates a green shaded area covering the whole space within the white lines. If an object enters the green area, the computer vision system will recognize the spot as occupied. If the driver decides to take up two parking spaces, it's possible that computer vision system will falsely label the two spots as unavailable.

Another social constraint to be considered is the installation ease and use of the Park Shark System. The system should be easy to install and mount onto the wall or ceiling, depends on the customer's preference. The team will try to make the system as wireless as possible. This way, the customer will not have to worry about installing any additional wiring in the garage to power up the system or for communication purposes. The team should also figure out a way to make the system easy to activate.

4.5 Political Constraints

Since the Park Shark system is meant for parking garages and not for government use, there doesn't seem to be any political constraints. Perhaps the only political constraint the team must take into consideration while designing the system are the standards that must be followed. For example, there are certain standards that one must follow when soldering a circuit board or writing a code.

4.6 Ethical Constraints

While designing the Park Shark system, ethical constraints identified by the IEEE code of ethics should be taken into consideration. Any design parameters that violate the code of ethics should not be considered. The Park Shark team will not implement any components that place the customer or drivers in harm's way. Safety will be one of the team's highest priorities. The team also needs to make sure the system provides the most accurate information possible. The system should not provide false information about the parking space availability in the parking garage.

4.7 Health Constraints

The team should ensure that the Park Shark system does not impose any health hazards. The team should conduct further research on the potential effects some of the electrical components can impose on the drivers. For instance, the safety of wireless communication, such as Bluetooth, should be further researched due to

its use of electromagnetic waves. The ultrasonic waves transmitted by the ultrasonic sensors have no negative impact on humans and it doesn't fall within a human's hearing range. Also, the components that the team decides to implement in the Park Shark system should abide by the RoHS (Restriction of Hazardous Substances) standards. By doing so, the impact on the environment itself will be minimized.

In addition, the team should also look out does potential health concerns imposed by the power supply. Currently, the Park Shark team is thinking about using batteries as the main power supply for the sensor modules. There are a variety of batteries available on the market that differ in chemical composition. Some batteries are known to consist of toxic chemicals, such as nickel and cadmium, which are harmful to both humans and the environment. Therefore, the team will decide upon a power supply that does not possess any toxic chemicals that could potentially cause harm to the drivers or the environment.

4.8 Safety Constraints

One of the main constraints the Park Shark team must address is safety. One of the benefits the Park Shark system has to offer is that the customer has the freedom to place the sensor and camera modules wherever they please. For example, the two possible places to mount the sensor modules are the wall and the ceiling. If the customer decides to mount the sensor module onto the ceiling, there's a slight chance that it can fall onto somebody or a vehicle if not properly secured. This could result in a possible lawsuit for damage of private property. Obviously, the team would like to prevent this situation from occurring. To minimize the risk of such an incidence occurring, the sensor module will be built to be lightweight and it should be tightly secured to the ceiling. By making the sensor module lighter in weight, the chances of the device falling due to weight are lowered.

Another safety issue the team must address is privacy. Since the Park Shark system will be utilizing Wi-Fi and Bluetooth communication, it's possible for hackers to infiltrate the system. It's possible for the hackers to override the system and use the cameras for their own purposes. Also, a website and mobile application will be created for the Park Shark system which could potentially be infiltrated as well through the use of the IP address. Therefore, the Park Shark team will make sure to abide by proper security protocols to ensure the privacy of the drivers, website users, and mobile app users.

4.9 Manufacturability Constraints

There are various manufacturing constraints that can have an impact on the design of the Park Shark system. One manufacturing constraint is the availability of a select component. It's possible that a select component won't be in stock until much later in the year. If the part isn't in stock, then we would have to look for another part that can replace it. Besides availability, the team should note whether the component is sold in bulk or single units. It's possible that vendor might not sell the part as a single unit and the team wouldn't want to spend money on buying a

bulk. That would be a waste of money. Another constraint is the shipping and delivery time for products purchased online. It's important to order parts early in the semester or even the one prior in order to receive the selected components on time. While purchasing items online, it's also important to make sure the vendor/supplier is trustworthy. The team wouldn't want to receive a malfunctioning component or never receive one period. That would result in a waste of time and money. Therefore, the Park Shark team will make sure all the items being used for the system design are available for purchase, keep track of shipping dates/times, and purchase only from reliable websites.

4.10 Sustainability Constraints

One of the specification requirements for the Park Shark system is that the product must be able to operate for at least year. This goal can be achieved by carefully selecting the power supply for the sensor system and selecting electrical components that offer low power consumption. The operation time of the device depends on the power consumption and the power capacity of the selected power supply. Preferably, the power supply selected for the sensor modules should be able to supply enough power for at least a year without requiring a battery change. By doing so maintenance time will be lowered as well as the maintenance cost.

To extend the lifespan of the system overall, the team should take note of the typical lifespan stated for each component. Ideally, each part should have a lifespan that exceeds more than 2 or 3 years. Creating a system that needs to be replaced every year or less is unappealing. Such a system would simply be too expensive in the long run compared to a system that is guaranteed to operate for more than 10 years. Not only is it a waste of money, it would also be a waste of time since the old system would have to be uninstalled and the new one would be reinstalled. Another parameter to take into consideration is the components operating temperature. Placing a component in an environment that experiences a temperature that doesn't fall within the component's operating range will lead cause the device to malfunction, hence, shortening its life time.

For easy maintenance, the components selected for the computer vision system and sensor system should be easily replaceable and the overall design should be kept simple. This would be beneficial to the customer because instead of having to purchase a whole new system, the customer would be able to easily replace the malfunctioning component in the device. The team should make sure that the selected parts are sold by multiple vendors and take note of the manufacturing lifetime of each part. Ideally, the manufacturing lifetime of each part should be at least 5 years. This way the customer can be reassured that modifications and part replacements will be possible five years from now.

4.11 Testing/Presentation Constraints

One of the main constraints the Park Shark team must address is how the system will be tested and demonstrated. Unfortunately, the team does not have the funds necessary to implement the Park Shark system for the entire parking garage. The size of the parking garage would require the installation of multiple cameras and

sensor modules to cover all the available parking spaces. Therefore, the team decided to conduct a demonstration of the Park Shark system in a small section of parking garage C, most likely on the first floor. This will allow the team to test and demonstrate the functionality of the computer vision system in conjunction with the sensor system.

Another concern the team must address is Wi-Fi connectivity in the parking garage. The UCF Wi-Fi connection in the parking garage is known to be faulty and almost nonexistent. One of the primary features of the Park Shark system will be the website and mobile application which informs the driver about the parking space availability. Thus, internet access will be necessary for the Park Shark system to upload and update the parking space statuses on the website and mobile application. For testing purposes, the team thought about the possibility of using a phone's hotspot capability to provide the Park Shark system with a temporary internet access point.

5.0 Related Standards

This project will take advantage of a number of existing standards in order to accomplish its goal. Standards such as wireless communication will dictate how the product will be able to operate and communicate. Other standards such as soldering standards will dictate how the project will be assembled and will help ensure that the assembly of the product will exceed a certain measure of quality and durability if the standards are properly followed.

5.1 Soldering Standards and Guide

The first standards in consideration are the standards on soldering. This will ensure that the components attached to the product's circuit board will remain functional over time for as long as possible, rather than poor solder joints cracking or corroding during the device's operational lifetime.

5.1.1 NASA Standard

For this project, we will abide by the soldering standard provided by NASA (National Aeronautics and Space Administration). The standard, known as NASA-STD-8739, describes in extreme detail the requirements for reliable soldered connections. As our project will require the soldering of several components to meet our project requirements, we will briefly discuss the standard and relevant sections.

The first relevant section that will be upheld is section 6 of the standard document which details the required facilities, equipment, materials, and parts to perform a soldering job. All soldering must be done in a clean environment where the chance of external object or debris cannot interfere with the soldering. The area is required to have a no smoking, eating, or drinking policy. The area must have proper ventilation, as well as proper safety and emergency equipment such as eye protection and fire extinguishers. The equipment required to perform soldering includes a functional and clean soldering iron, means to control the temperature such as conductive-type irons and resistance-type soldering. A solder pot or brass sponge must be present at the soldering workspace to store the solder when cooling. The use of a vice and clamps should be present to hold the components in place for ease of soldering. A wide degree of stripping tools will also be required in the advent of an incorrect solder application. Bending tools such as small non-conductive tweezers and pliers should be present as well. Mechanical stripers (such as a solder pump), Thermal strippers, and Chemical Strippers should be present at any soldering workspace. Inspection Optics, such as a stationary magnifying glass should also be present at the workspace to assist with the soldering operation. Solder and Liquid Flux should also be available in large quantities at the workspace to allow for efficient work. A wide selection of Solvents and Cleaners should also be present, such as Ethyl Alcohol and Isopropyl Alcohol.

The area selected for this project that meets these requirements will be the TI Innovation Lab located at UCF in the Eng2 building. This area provides dedicated work spaces for soldering, as well as providing other required tools and item

describes in the above paragraph. The lab provides all soldering tools, solder, cleaners, and safety equipment which meet the requirements set forth in the standards document.

The next relevant section is chapter 11, which discusses the process for soldering to terminals, which will be used heavily in this project. There should be no movement between the terminals before soldering begins. If any movement occurs, they should be replaced and secured before further soldering occurs. The solder coverage should be applied around to conductor and over the termination area. Pressurized air will be strictly prohibited for any cooling purposes. All connections shall be cooled at room temperature only. The solder will be cleaned after each soldering operation. Inspection of each connection to ensure quality will be required.

5.1.2 SparkFun Guide

A brief soldering guide provided by Spark Fun was also used for reference when writing this section. The guide details the required tools and environment similar to the NASA standard discussed above. The guide included a section detailing the soldering process that will extremely helpful for this project as members are not extremely familiar with the soldering process.

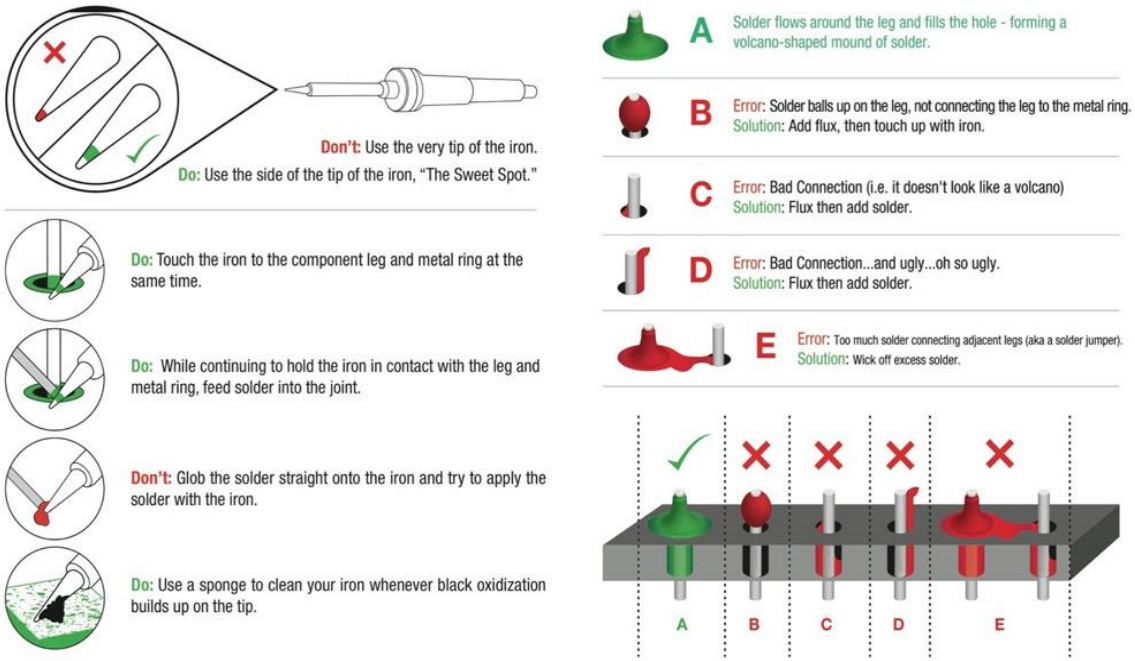


Figure 16: Proper soldering technique. Reprinted with permission from SparkFun.

The first and most important step, as discussed in the standard, is to secure the component with vices to ensure a stable base. After heating up the iron to a heat around 325 to 375 degrees Celsius. Next put a light layer of solder over the soldering iron. This process is known as tinning. This process assists with the conductivity and also makes the soldering process quicker. Next place the iron on

the area you want to attach your component to heat it, then apply the solder. After applying a decent amount, remove both the solder and the iron. An image detailing these steps and showcasing proper and improper soldering is shown below in Figure 16.

5.2 Programming Standards

Although coding standards do not necessarily affect the compilation of the code, it is important in many other ways. When coding standards are implemented correctly, the code's maintainability is much higher. It also makes it easier for the developer to debug code and to add new code. The largest advantage to following a coding standard is the advantages that it brings when implementing integrations of other components. The coding standards that the group will be enforcing with the Park Shark system are formatting standards, and naming standards. These will help with overall code readability.

5.2.1 Formatting Standards

Formatting is a key component when it comes to coding standards and the main ones this section will look at are braces, indentation, line length, whitespace, and comments. These standards dictate how the code looks, its spacing and indentation will determine how quickly readable the code is at a glance. If the formatting is too cramped, then the code will be much harder to read than one spaced out with proper tabs and brace standards.

5.2.1.1 Braces

Braces are used with if, else, for, do, and while statements, even when the body is empty or only contains a single line statement.

Examples:

```
// This is acceptable
if (hour == 10)
{
    counter += hour;
}

// This is not acceptable: braces are used even on single line statements
if (hour == 10)
    counter += hour;
```

5.2.1.2 Indentation

Each time a new block or block-like construct is opened, the indent increases by a tab. When the block ends, the indent returns to the previous indent level. The indent level is kept consistent throughout the block. The following code snippets are examples of different kinds of indent levels, to show their effect on how readable the code is.

```
// This is acceptable
for (int i = 0; i < 10; i++)
{
    for (int j = i; i < 10; i++)
    {
        biggy *= j;
        small += j;
    }
}

// This is not acceptable: indentation by spaces
for (int i = 0; i < 10; i++)
{
  for (int j = i; i < 10; i++)
  {
    biggy *= j;
    small += j;
  }
}

// This is not acceptable: no indentation
for (int i = 0; i < 10; i++)
{
for (int j = i; i < 10; i++)
{
biggy *= j;
small += j;
}
}
```

5.2.1.3 Line Length

Line length should not exceed 100 characters. Any line that would exceed this limit must be line wrapped.

5.2.1.4 Whitespace

There should be single blank line between consecutive members or initializers of a class. Vertical whitespace is also permitted as needed to organize logical subsections.

Horizontal whitespace is required between reserved words, like if, for, while, from an open parenthesis. There should also be a space after any comment line. Lastly horizontal alignment is never required.

5.2.1.5 Comments

Any form of comments is permitted with the exception of comments with excessive whitespace.

5.2.2 Naming Standards

All naming standards will follow the upper camel case (initial uppercase letter), lower came case (initial lowercase letter), or constant case (all uppercase letters).

Table 22: Definition of naming standards for type names in programming.

Type	Standard	Name Example
Class	Upper Camel Case	ExampleClass
Method	Lower Camel Case	exampleMethod
Constants	Constant Case	CONSTANT_CASE
Parameters	Lower Camel Case	exampleParameter
Variables	Lower Camel Case	exampleVariable

5.3 Power Supply Standards

The agencies responsible for the standards regards electrical safety are International Electrotechnical Commission (IEC) and the International Organization for Standardization (ISO). The IEC60065 standard applies to audio, video and similar electronics. This standard is meant to protect against fire, electric shock and injury. The IEC60950-1 standard talks about safety pertaining to technology equipment. This standard is applicable to battery-powered information technologies with a voltage that doesn't exceed 600V. Under this standard, three different classes of equipment are identified. Class 1 equipment protects from electric shock through insulation and grounding. All conductive parts are to be connected to a protective earth conductor. Class 2 equipment provides more protection from electric shock by using more insulation. Class 3 equipment operates from a Safety Extra Low Voltage (SELV). There are five types of insulation: operational/functional, basic, supplementary, double, and reinforced. For power supplies, the minimum insulation requirements are primary to secondary and primary to ground.

5.4 IEEE Standards

IEEE stands for the Institute of Electric and Electronic Engineers. IEEE is one of the largest professional organizations in the world and provides the definition of most of the standards used in electronics. These standards include standards for power supply, standards for transformers, and standards for communication such as wireless frequencies.

5.4.1 Wi-Fi Standards

Wi-Fi is based on is based in IEEE 802.11 standards. IEEE 802.11 is a set of standards for implementing wireless local area network (WLAN). Different Wi-Fi technologies pertain to a specific set of wireless standards such as 802.11,

802.11a/b/g/n, or 802.11ac. A comparison between these standards is shown below in Table 23.

Table 23: Summary of Wi-Fi IEEE standards.

IEEE Standard	Frequency	Max Data Rate	Range
802.11	2.4 GHz	2 Mbps	20 ft
802.11a	5 GHz	54 Mbps	25-75 ft
802.11b	2.4 GHz	11 Mbps	150 ft
802.11g	2.4 GHz	542 Mbps	150 ft
802.11n	2.4 GHz / 5 GHz	600 Mbps	175 ft
802.11ac	5 GHz	6.93 Gbps	230 ft

5.5 Bluetooth Standards

Bluetooth is a wireless communication standard operating in the 2.4 GHz frequency band. Unlike Wi-Fi, it is designed to prioritize low power draw over data throughput. Bluetooth began as IEEE Standard 802.15.1, but the Bluetooth SIG has taken over managing the Bluetooth standard instead of IEEE. Bluetooth Core specification dictates that devices can connect over a distance of no less than 10 meters. All of these standards are implemented by transceiver manufacturers into their chips, it is no up to the group to implement these standards. However, the relevant standards and their importance to the project are described below, as they are good criteria for deciding the right components to use.

5.5.1 Bluetooth 5.0

Bluetooth 5.0 is the latest version of the Bluetooth standard. Notable additions include the ability to quadruple the effective range at the cost of lower data throughput. It also reduces power consumption in general. These changes are highly attractive for the wireless parking sensor, allowing it to have longer battery life. Many manufacturers have been able to implement some Bluetooth 5.0 features into their previous Bluetooth 4.2 devices, allowing some degree of upgradeability despite not being fully compliant with the whole Bluetooth 5.0 feature set. The Nordic Semiconductors nRF52832 is one example of this; it was formerly a Bluetooth 4.2 chip upgraded to support the power savings functionality of Bluetooth 5.0, as well as support for the Bluetooth Mesh standard.

5.5.2 Bluetooth Low Energy (BLE)

Bluetooth Low Energy is a side standard developed to integrate with Bluetooth 4.0, targeting low power devices, such as devices powered by coin cells. Bluetooth Low Energy is designed to have similar range to standard Bluetooth while having far less power consumption. Compatibility with Bluetooth Low Energy devices is

integrated into the Bluetooth 4.0 standard, which carries over to Bluetooth 5.0 as well.

5.5.3 Bluetooth Mesh

Bluetooth Mesh is a separate standard from the main Bluetooth standards that allows multiple devices to connect to each other, through each other. This means that a device can reach another Bluetooth device outside of its sensor range by sending its data through other Bluetooth devices along the way. Bluetooth Mesh is ideal for this project's application, as it will allow much more freedom in the placement of the wireless sensor modules around the garages, allowing them to communicate back to the camera module through each other rather than requiring all of them to be able to connect to the computer vision camera module directly. This will greatly increase the range that the wireless ultrasonic sensor modules will be able to be spread out around a parking garage or parking lot.

6.0 Design

The design section is split into two: one section for the wireless sensor to be placed at parking spots, and another section for the computer vision-based camera sensor that will oversee multiple parking spots.

6.1 Sensor System Design

This section will detail the design of the software and hardware used in the sensor system. In this section, we will give a brief overview of the system, discuss the individual components that make up the sensor module, and then detail the design and implementation of this system.

The sensor system will utilize an nRF52832 module as its central computing device and communication device. By assigning the nRF52832 module as the main processing unit in place of a typical microcontroller board, we're able to save money and power consumption. The nRF52832 module will oversee the communication between all the sensor modules in the system and keep track of their individual statuses. The module will then send the data it has collected to the computer vision system. The sensor module will utilize an ultrasonic sensor to detect vehicles in the parking spaces. A pack of 3 AA batteries will be used as the power supply for the whole sensor module.

6.1.1 Hardware Block Diagram

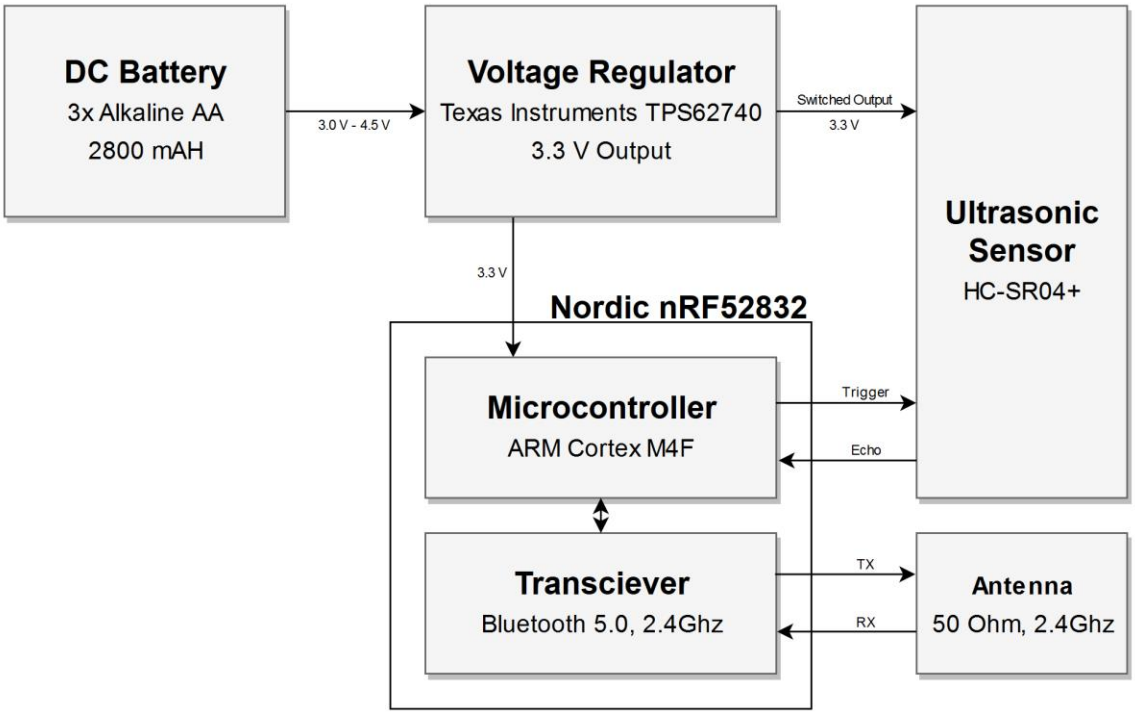


Figure 17: Hardware block diagram for wireless sensor.

6.1.2 Microcontroller + Bluetooth Transceiver

Most of the technical details for the microcontroller, contained within the Nordic nRF52832 Bluetooth transceiver chip, are covered in Section 3.7. The Bluetooth chip serves as the microcontroller for this system as well as the transceiver. It will calculate the input from the ultrasonic sensor and transmit this information wirelessly back to the parking system server via Bluetooth link to the computer vision camera module. The chip has a sleep mode power draw of only 2.2 microamps, so to maximize battery life it will spend most of its time in sleep mode, only waking up at certain intervals to perform sensor checks and transmit data. This will help ensure that the system has as long of a battery life as possible. The chip will be powered by 3.3 V input from voltage regulation components and will have GPIO pins connected to the ultrasonic sensor, the voltage regulator (to switch off the ultrasonic sensor when not needed), and an LED to indicate status when enabled in configuration. The LED will normally be disabled, because it will have a relatively high power draw compared to the other components used in the sensor.

6.1.3 Ultrasonic Sensor

One of the main components in the sensor module is the ultrasonic sensor. The ultrasonic sensor, shown in Figure 18, will be used to detect vehicles in a certain amount of parking spaces. Compared to other sensors that were considered, the ultrasonic sensor offered the most range and ease of use and met all of the other criteria.

Figure 18: Ultrasonic sensor module. Reprinted with permission from SparkFun.

The chosen ultrasonic sensor is the HC-SR04. This ultrasonic sensor has a 5V input requirement, which is above the range of the Nordic nRF52 chip's voltage range. Therefore, to make use of this sensor, there would be some challenge in designing a power supply for the circuit to efficiently handle supplying 5 volts as well as 3.3 volts. Luckily, the group discovered that there are lesser known newer revisions of the HC-SR04, which are known as the HC-SR04+ and another known as the HC-SR04-P. These revisions lower the idle current draw of the sensor and also increases the tolerable input voltage range to 3.0V – 5.5V. This allows the

circuit and power design to be much simplified and allow the device to run from lower voltage sources without using wasteful voltage converter circuits to also supply the 3.3V for the Bluetooth board. These newer revisions are not as readily available in as many stores as the standard HC-SR04, but they are identifiable by looking for the “HC-SR04+” or “HC-SR04-P” print on the back of the sensor as well as differing layouts of the three ICs on the chip.

The specifications of the ultrasonic sensors are listed below in Table 24.

Table 24: Comparison of HC-SR04 vs HC-SR04+ specifications.

Specification	HC-SR04	HC-SR04+
Operating Voltage	4.5 to 5.5 V	3.0 to 5.5 V
Operating Current	10 to 20 mA	10 to 20 mA
Quiescent Current	4 mA	3 mA
Frequency	40 kHz	40 kHz
Range	2 cm to 400 cm	2 cm to 450 cm
Resolution	3 mm	3 mm
Temperature Range	-15 to 70℃	-15 to 70℃
Dimensions	43 mm x 20 mm x 15 mm	43 mm x 20 mm x 15 mm
Weight	8.5 g	8.5 g
Sensing Angle	30˚cone	30˚cone
Angle of Effect	15˚cone	15˚cone
Trigger Input Signal	10µs TTL pulse	10µs TTL pulse
Echo Output Signal	Output TTL PWL signal	Output TTL PWL signal
Price	$3.95	$1.44

6.1.3.1 Working Principle of Ultrasonic Sensor

The ultrasonic sensor module consists of two ultrasonic transmitters, a receiver, and a control circuit. The ultrasonic transmitter transmits high frequency ultrasonic waves. If an object is present, the ultrasonic waves traveling through the air will hit and bounce off the target. The reflected ultrasonic waves, or echo, will be collected by the receiver. The control circuit will then process and calculate the distance between the object and itself by measuring the return time of the reflected signal. Overall, the sensor is easy to set up since it has only four pins: VCC, Trig, Echo, and GND

To start up the device, a high 10µs pulse must be supplied to the Trig pin of the sensor module. Once the Trig pin receives this pulse, the ultrasonic transducer will transmit an 8-cycle burst of ultrasonic waves at a frequency of 40 kHz. When the receiver of the sensor module detects reflected ultrasonic waves, the Echo pin will be set to high whose pulse width corresponds to the distance. The distance is then calculated by measuring the width of the echo pulse. The following formula is used to calculate the distance:

$$\text{Distance (inches)} = \frac{\text{Width of the Echo Pulse (µs)}}{148}$$

It is suggested that the measurement cycles should be set to occur a value that exceeds 60ms.

6.1.3.2 Programming for the Ultrasonic Sensor

Using the principle discussed in 6.1.3.1, the programming of the microcontroller to handle the ultrasonic sensor is straightforward. The VCC pin of the ultrasonic sensor is supplied from the voltage regulation components, so the microcontroller does not control that pin. However, the ultrasonic sensor should not be left powered all the time as it has a fairly high current draw (3mA) even when sitting idle and sending no ultrasonic pulses. Therefore, one GPIO pin of the nRF52 will be used to control power switching for the ultrasonic sensor.

The Trig pin of the ultrasonic sensor will be connected to an output GPIO pin of the nRF52. When it is time for the microcontroller to use the sensor, the microcontroller will set the Trig pin to High for at least 10 microseconds, then drive it back to Low. Then, it will wait until it receives a signal coming from the Echo pin of the ultrasonic sensor, which will be connected to a GPIO input pin of the microcontroller. It will sample the current clock time when the Echo pin is driven High, then sample the clock time again when the Echo pin is set back Low. The two time values will be subtracted in order to obtain the width of the echo pulse, and divided by 148 to obtain the distance reported by the ultrasonic sensor. This distance will be compared to a configured threshold value. If the distance exceeds this threshold, then the nRF52 will determine that the parking spot is unoccupied and will inform the parking management server when the next wireless activation cycle occurs.

6.1.4 Power Supply

In this section, several battery types that can supply a constant output voltage to the unit will be considered. For the sensor system, our goal is to create a sensor module that can operate for at least a year before requiring a battery change. To select a long-lasting power supply, we must first identify the consumption characteristics of each active element in the sensor module and plan activation intervals in order to ensure that the battery life will last for as long as possible.

For the sensor system, it was decided that the sensor units will be powered by a DC power supply. So far, the components in the sensor unit that will require DC power are the nRF52832 module and the ultrasonic sensor. The nRF52832

module requires a supply voltage of 3.3V and the supply voltage range for the ultrasonic sensor is between 3 and 5.5V. The most important specification to consider regarding these components is the amount of current they draw during activation. This factor plays a key role when deciding on a power supply because battery life depends on battery capacity and current draw. Therefore, the team must first figure out the sensor module's total current draw over a 24-period when programmed to activate at given intervals. In Table 25, the current draw for each of these active components in the sensor module is listed. These values along with others will be taken into consideration while calculating the total current draw by the device.

Table 25: Current draw for each active element in the sensor module.

Active Components	Active Current (mA)	Quiescent Current (mA)	Power (mW)
HC-SR04+ Ultrasonic Sensor	15	2	75
nRF52832 Module	5.4	0.0003	16.2
TPS62740 Step Down Converter	100	0.00036	150
Total	20.457	2.0008	241.2

6.1.4.1 Estimating Sensor Current Draw Per Hour

For the wireless sensor module, the key to having a long battery life will be activating all of the electronic components, such as the Bluetooth microcontroller and the ultrasonic sensor, only at specific intervals. When the microcontroller can spend the majority of the time in sleep mode, the power consumption will be very low, in the range of tens of microamps or less. The downside to having the components in sleep mode most of the time is that the less often the system wakes up to check the status of a parking spot, the less responsive the live updating of parking status will be, increasing delays in the web GUI showing an accurate count of available parking spaces. So, the group must come up with a method to balance these two design requirements. The group decided that up-to-date status information is less important the less full the garage is, so when the garage is not at near capacity, the sensor checking interval can be reduced to save battery life, but at high occupancy the sensor can wake up to check a parking spot status more often to provide frequent updates on highly contested spots.

To come up with a scheme for what intervals the sensor should activate on, data was gathered from the current UCF parking occupancy system over one day to chart the occupancy of all garages over time. The data was sampled by checking the UCF garage occupancy web app and recording the data every 30 minutes over 24-hour period. The results are displayed in Figure 19. This data will be used to estimate the power consumption requirements of the device over one day. Because it is an estimation, this data will not be strictly adhered to as its purpose is only to demonstrate the conditions of an average day to estimate power

consumption. Another reason the data will be adjusted is that the current UCF parking occupancy system is prone to inaccuracy.

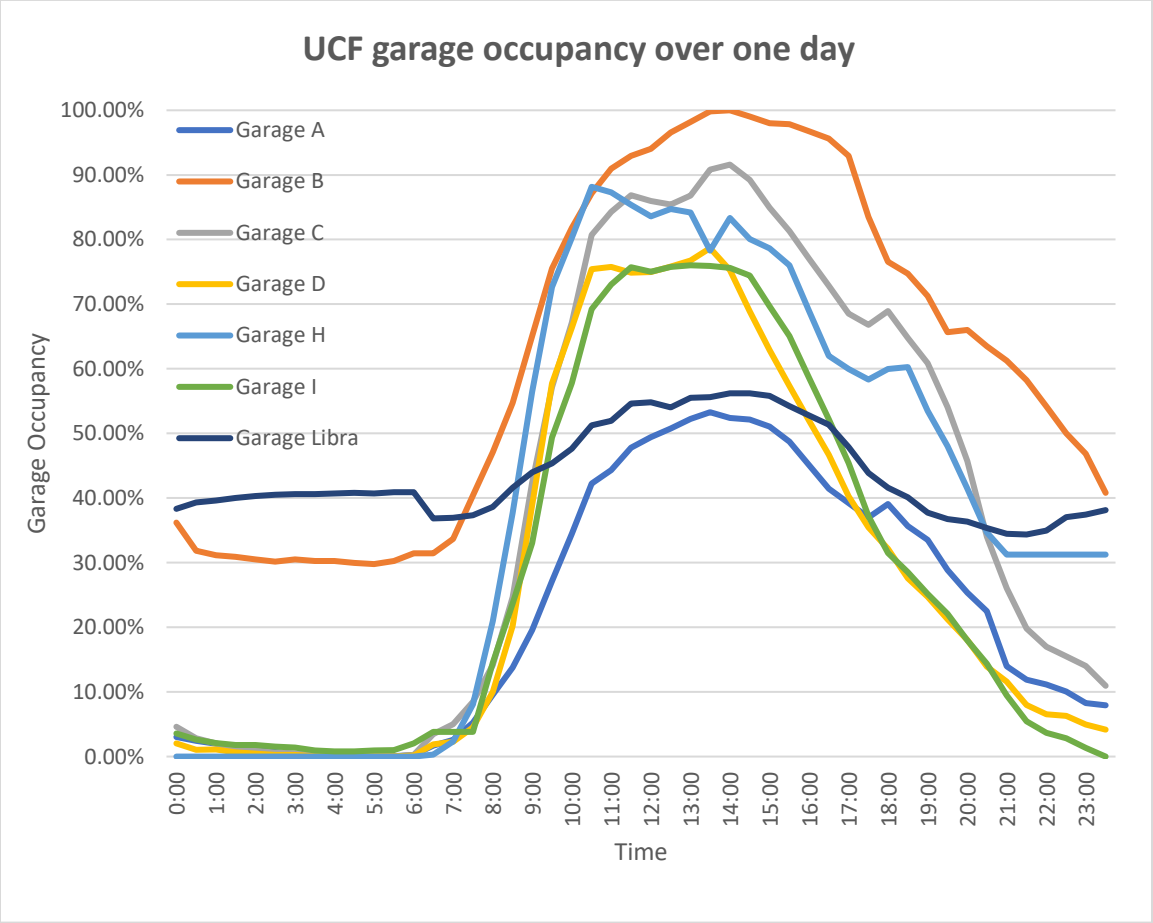


Figure 19: Gathered data on UCF garage occupancy vs. time.

Notably, the gathered data presents some anomalies, such as Garage H being stuck at 30% occupancy at night despite overnight parking being disallowed at that garage. The actual garage occupancy at that time is most likely closer to 0%, indicating that the open spots reported by the current UCF parking monitoring system can be off by 30% of total garage occupancy or more. Additionally, the occupancy is not often shown to be 100%, even though the garages are usually completely full for several hours of the day. Due to these observations and inaccuracies compared to the real-world occupancy, the gathered data must be adjusted to represent the real-world garage occupancy more closely. This adjusted data line is presented in Figure 20. The model to estimate power consumption will be based on the Garage B data, as its overnight occupancy allows for designing the circuit's power specifications to handle a busier average day than a garage with no overnight occupancy. Personal experience from the group shows that Garage B is usually completely full at around 10:00 AM, with cars circling around waiting for any spots to open. The gathered data from Garage B has been scaled to show this.

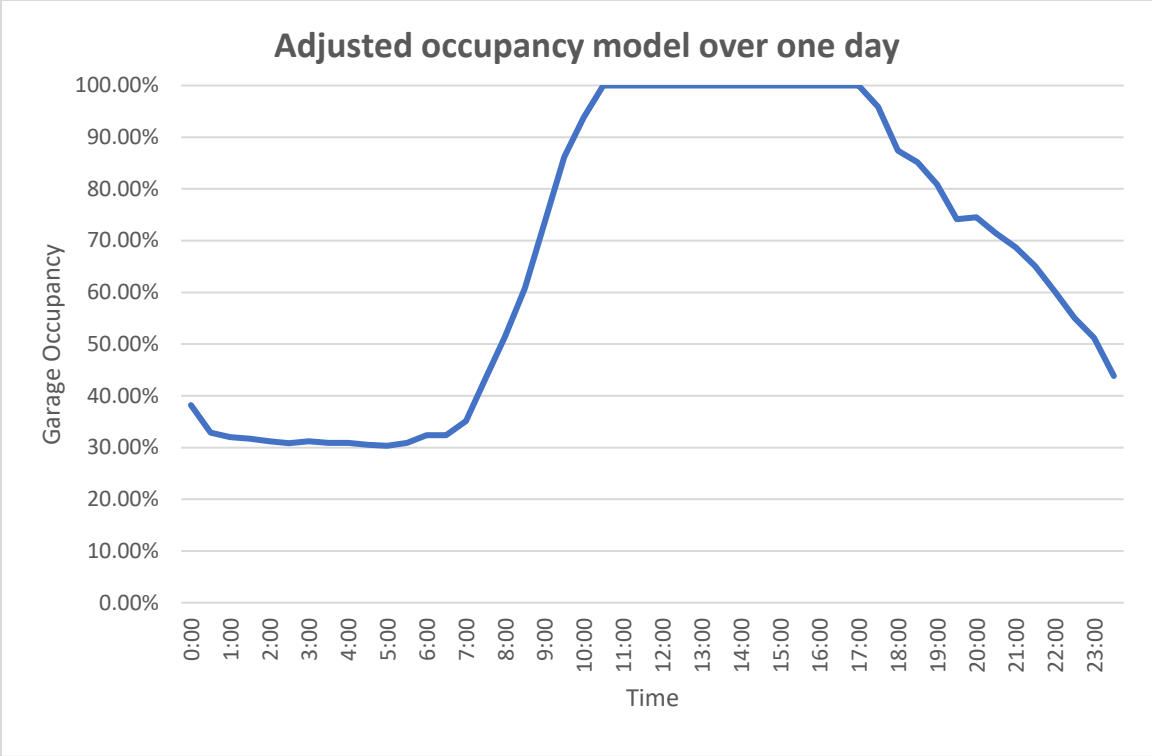


Figure 20: Adjusted data model for garage occupancy vs. time.

With this baseline model for the occupancy characteristics of an average day in a parking garage, the group can now establish interval levels to reduce the time the sensor is active in order to extend battery life. There will be four interval settings depending on the garage's current occupancy. When the occupancy falls below a certain threshold, the sensor will go into the next interval level, which means it will wake up to check the parking spot less often until the garage occupancy goes above the top threshold.

Table 26: Proposed intervals for sensor checks

Interval Level	Occupancy >	Occupancy <=	Interval Length	Activations Per Hour
1	95%	100%	30 seconds	120
2	85%	95%	1 minute	60
3	70%	85%	5 minutes	12
4	0%	70%	10 minutes	6

Now, the interval settings in Table 26 will be applied to the data in Figure 20 in order to calculate the average sensor activations per hour. From this, the average consumption per hour of the sensor module can be estimated, and then a battery can be selected in order to maintain the estimated power requirement for at least 1 year. The calculations to obtain the average sensor activations per hour are given in Table 27.

Table 27: Calculation of estimated sensor activations per hour.

Interval Level	Hours per day spent at this interval	Total activations at this interval
1	7.5	900
2	2	120
3	2.5	30
4	12	72
	Total per day:	1122
	Average per hour:	47

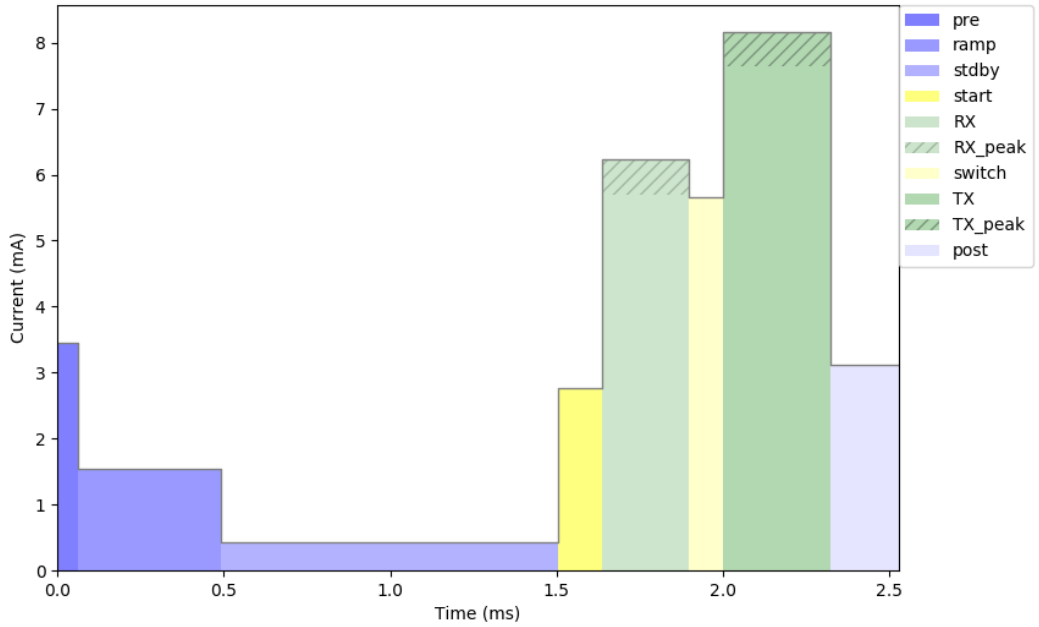
The occupancy thresholds and interval length can be adjusted to change the average activations per hour to extend battery life. The group decided that the interval settings above provide the best compromise for up-to-date status reporting and battery life. It is interval level 1 (nearly full to full garage occupancy) that will account for most of the total sensor activations, but if the sensor spends only some of its time at this interval level then power draw can be reduced.

6.1.4.2 Current Draw Per Component

With the estimated 47 activations per hour given by Table 27, the group can estimate how enabling and disabling certain components at certain intervals will affect battery life.

6.1.4.2.1 Bluetooth Chip

The first component to be examined is the nRF52830 Bluetooth transceiver and microcontroller. While the specifications for the Bluetooth module define the peak power draw when transmitting/receiving and the sleep mode power draw, the manufacturer Nordic provides a much more detailed power draw calculator for the nRF52 platform on their website (https://devzone.nordicsemi.com/power/). This calculator breaks down the estimated power draw at all stages of the nRF52's operation, and allows adjusting parameters for wireless activity interval, data packet size, and transmit power. The group used the slowest transmit interval (4 seconds) and the highest TX power (4dB) and packet size (32 bytes, likely far more than are needed to transmit the status information) in order to design the system for a higher load than is necessary. The results of the Nordic power calculator are presented in Figure 21. The resulting average power consumption is only **3.7 microamps**. With a power consumption this small, the group decided that there is no reason to turn off the Bluetooth chip for intervals longer than 4 seconds or adjusting it to follow the earlier calculated sensor check intervals. As a result, the Bluetooth connections will be easier to maintain as the sensor can transmit more often. Therefore, in terms of power calculations, the Bluetooth chip will be considered mostly negligible. It is likely that the ultrasonic sensor and the voltage regulation components will draw much more power than the Bluetooth module, so most of the consideration will be for those components.



Stage	Description	Time (ms)	Length (us)	Avg. current (mA)	Peak current (mA)
pre	Pre-processing	0.0	61	3.4	
ramp	Standby + HFXO ramp	0.1	430	1.5	
stdby	Standby	0.5	1014	0.4	
start	Radio startup + CPU	1.5	133	2.8	
RX	Radio RX	1.6	261	5.7	6.2
switch	Radio switch	1.9	102	5.7	
TX	Radio TX	2.0	323	7.6	8.2
post	Post-processing	2.3	205	3.1	
	System On IDLE	2.5	3997.5 ms	2.0 uA	
Total			**4000.0 ms**	**3.7 uA**	

Figure 21: Calculated power consumption of Bluetooth chip.

6.1.4.2.2 Ultrasonic Sensor

As explained in Section 6.1.3, the HC-SR04+ Ultrasonic Sensor has a quiescent current draw of 4 milliamps. This is the power draw when the sensor is doing nothing other than being powered by its VCC input, sending no pulses or signals. This 4 milliamp current draw is much higher than the Bluetooth chip's 3.7 microamp current draw. Battery life is expected to be very low with a constant 4mA current draw, and will likely only last for a couple of days. Therefore, for maximum battery life, the ultrasonic sensor must have its VCC input cut off in order to approach the 1 year desired battery lifetime. The ultrasonic sensor will be disabled and re-enabled according to the interval levels defined in Section 6.1.4.1. When the ultrasonic sensor is unpowered for most of the time the wireless sensor is active, then the power draw will be much lower than 4 milliamps.

First, the group must determine how long the sensor should be active until VCC is cut off. As the speed of sound is 343 meters per second, we only need the sensor to be active for the amount of time for sound to travel only a couple of meters, as this will be the sensor's range to the car it is trying to sense. The group decided based on this requirement that the sensor should only need to be active for 100 milliseconds before its power is cut off. This should be more than enough time for

the sound waves sent by the ultrasonic sensor to be returned if the distance is only a couple of meters., as sound can travel 34.3 meters within 100 milliseconds.

While the HC-SR04 has a quiescent power draw of 4 milliamps, it draws around 16 milliamps when it is sending a soundwave pulse and waiting for the wave to return. For estimation purposes and to prepare for a higher power draw than is needed, the group will assume that the HC-SR04+ will be drawing the full 16mA for the entire 100 millisecond window that the sensor will be powered. Using these values, the current draw per hour of the HC-SR04+ will be around **20 microamps**. This power draw is still quite higher than the current draw of the nRF52 Bluetooth module; however, it is a much lower value compared to 4mA and the battery life with this draw should be expected to be much longer. Most importantly, for this power saving method to work, the HC-SR04's VCC must be powered by a switchable power input controlled by the nRF52's microcontroller.

6.1.4.2.3 Voltage Regulation

The voltage regulator module, or VRM, is the last significant power consuming component of the circuit. It must convert a varying range of voltage input to the required 3.3 voltage output to power the nRF52 chip and the HC-SR04 ultrasonic sensor. Without it, most forms of battery input will not be possible as battery output voltage tends to vary with its capacity. A voltage step-down regulator must be chosen that will provide the most efficient power output for the sensor in order to maximize battery life. The two most common kinds of step-down voltage regulators are the Low DropOut regulator (referred to as LDO) and the Buck step-down regulator.

LDO regulators are fairly cheap, and do not require many supporting components (if any) other than the LDO regulator itself. This means that the LDO will have a very small circuit board footprint and component cost compared to a Buck regulator. However, the method LDO uses to step down the voltage is power dissipation. This means that much of the power going into the LDO will be radiated off as heat. This has many disadvantages. One disadvantage is the heat output, in an enclosed sensor module the heat will not have many opportunities to be radiated away from the sensor and circuit board. Most LDOs will shut off if the temperature reaches a certain threshold, this will effectively power down the entire sensor. The second major disadvantage is the inefficiency. In the case of a 5-volt input being stepped down to a 3.3 volt output, the power consumption will not be able to exceed 3.3/5 or 66% efficiency. This means that about at least third of the power stored in the batteries will be wasted as heat. This is a very unacceptable disadvantage for our application in terms of maximizing battery life. Efficiency will only increase as the input voltage drops approaching the output voltage. However, if the group was able to guarantee a voltage input close to the output voltage then there would not be much need for the voltage regulator at all.

The second kind of voltage regulator is the DC/DC Buck step-down regulator. This voltage regulator is known for being much more efficient at higher voltage differences than the LDO regulator. It used inductive components to convert the input voltage to a lower output voltage, transferring most of the power through the

output instead of radiating it as heat. The downside is that the circuit is much more complex, often requiring external capacitors, inductors, and even transistors to complete the circuit. This means it will have a higher board footprint and component cost. Additionally, the use of inductive components means that the circuit can generate some noise compared to the LDO. For the project application, however, the switching frequency of the Buck converter is several orders of magnitude less than the 2.4GHz Bluetooth wireless communication band, closer to the range of a thousand MHz instead. It should not significantly interfere with the Bluetooth RF circuit.

The group found a Buck converter tailor-made for the low power sensor's power draw characteristics. The Texas Instruments TPS62740 specifies a 90% efficiency rating at an output of 10 microamps. The wireless sensor will spend most of its time very close to this power draw as most of the components will be in a low power sleep mode. This is different than most Buck converters as they tend to have a 90% efficiency closer to the 1 milliamp output range. The TPS62740 also has a secondary switchable output that can be controlled by a microcontroller, allowing a convenient method to cut off power to the ultrasonic sensor instead of setting up a separate switching circuit. The group has chosen to use the **Texas Instruments TPS62740** as the circuit's voltage regulator, converting the battery input to two 3.3 volt outputs.

6.1.4.2.4 Status LED

There will be a status LED connected to the circuit, but in normal operation this LED will never be activated. It will only be activated when the sensor is configured to be in debug mode, in order to quickly be able to see what operating status the sensor module is in. When the LED is disabled, normal operation of the sensor module will be verified by checking for the module in the parking system's server rather than from the sensor module itself. This will ensure a long battery life, as the LED always being on would considerably shorten it.

6.1.4.3 Battery Comparison and Selection

With this information, the group is now able to select a battery that supplies the necessary voltage needed to power the device throughout the day and possibly an entire year. In order to select a battery for the sensor module, the overall voltage supply for the unit must be considered. The sensor module consists of an nRF52832 module and an ultrasonic sensor. The nRF52832 module requires a supply voltage of 3.3V. The supply voltage range for the ultrasonic sensor is between 3 and 5.5V. The team decided to pick 3.3V as the supply voltage for the ultrasonic sensor because the it's the same voltage required to power the nRF52 module. By doing so, the circuit can be simplified by utilizing one voltage regulator instead of one for each component. This will also lower the total cost of the module since it would eliminate the need to purchase two separate regulators. Based on this information, the supply voltage necessary to power the sensor module is about 3 to 4.5 volts.

Since the goal is to create a sensor unit that can last for at least a year without the need for a battery change, the selected battery would have to possess a high-

power capacity. The team would also like to choose a power supply that is user-serviceable to make increase maintenance ease. To select a long-lasting power supply, we must identify and compare the parameters of the most common battery types available on the market. Table 28, the voltage parameters for each battery type are listed.

Table 28: Battery voltage comparison chart according to chemical composition.

Chemistry	Nominal Voltage (V)	Cutoff Voltage (V)
Alkaline	1.5, 9	0.9, 4.8
Lithium-ion	3.6	3
NiMH	1.2	1
NiCad	1.2	1
Lead-Acid	2.1	1.75

*Permission granted by SparkFun.

Compared to all the other battery types, alkaline batteries offer a more economical solution. The team selected alkaline batteries because they are low in cost and they are easily attainable which increases maintenance ease. Alkaline batteries are also meant for lower current drain devices.

The two most important parameters to take into consideration when selecting a battery is its nominal voltage and its battery capacity. The nominal voltage is the average voltage the battery outputs when charged which depends on the battery's chemical composition. The battery capacity is the amount of power it contains rating it by the amount of current it can supply over a period of one hour (ampere-hours). The battery capacity necessary for the device can be calculated knowing the total current drawn by the active elements in the sensor module and the desired battery lifetime.

$$\frac{\text{Battery Capacity (mAh)}}{\text{Current Draw (mA)}} = \text{Battery Life (hours)}$$

In this case, the estimated total current drawn by sensor module was calculated to be about 50μA and the desired battery life is at a year or 8760 hours. By plugging these values into the previously mentioned formula, the necessary battery capacity is 438 mAh. This value exceeded the team's expectations.

$$\frac{\text{Battery Capacity (mAh)}}{0.05\ \text{(mA)}} = 8760\ \text{(hours)}$$

Now that the battery capacity has been calculated, an alkaline battery size can be selected based on their battery capacities. The battery capacity for each alkaline battery size available is shown in Table 29.

Table 29: Battery capacity according to alkaline battery sizes

Battery Size	Battery Capacity (mAh)
9V	570
AAA	1150
AA	2870
C	7800
D	17000

*Permission granted by Battery University.

Based on these calculations and Table 28, the battery size that offers a battery capacity that exceeds the estimated value are AA batteries. The team decided to pick AA batteries due to its small size, easy attainability, and low cost. Also, by providing a battery capacity of 2870 mAh, it's possible that the device will be able to operate for at most 57400 hours! How many AA batteries would be required to power the sensor module? Considering the necessary supply voltage for the sensor module is between 3 to 4.5V, three AA batteries could be used to fulfill this requirement. The nominal voltage of an alkaline cell is 1.5V, so the combination of 3 AA batteries would provide the necessary 4.5V (3 * 1.5 = 4.5V).

6.1.5 System Housing

The housing for the sensor module system will be described in this section. In accordance with the project specifications and requirements detailed in section 2, the dimensions of the entire system should not exceed 4 x 3.7 x 3.1 inches and the weight should not exceed 3 lbs. This way, the sensor module will not require industrial hardware to mount it. For demonstration purposes, the small size and weight might allow for the usage of Velcro strips to mount the device onto the wall or ceiling.

The housing for the ultrasonic sensor will be created using 3D printing. The Texas Instruments innovation lab located in the atrium at UCF provides free 3D printing which can be used to print the housing along. By 3D printing the housing, the team has the liberty to decide on the specific size parameters for the sensor module encasement. Most importantly, the housing will be free of cost. The design of the housing will consist of a cylindrical box with a small opening allowing for the ultrasonic sensor's transmitter and receiver to be exposed. The housing will also consist of a rubber material to wrap around the upper part of the box. This will be used as a safety precaution to protect the sensor device in case a driver pulling up into the parking space bumps the sensor with the car's front bumper. The diameter of the opening will have to big enough for the 30 degrees cone the sensor requires for operation. The design will also take into account the fact that the sensor device can be mounted either on the wall or the ceiling. This will require external mounting hardware to securely mount the sensor to the parking garage ceiling or wall. 3D printing also allows the use of lightweight materials such as nylon to create the box

which will allow for an extremely lightweight product that is also incredibly sturdy. The general idea for the sensor housing is shown in Figure 22 and Figure 23.

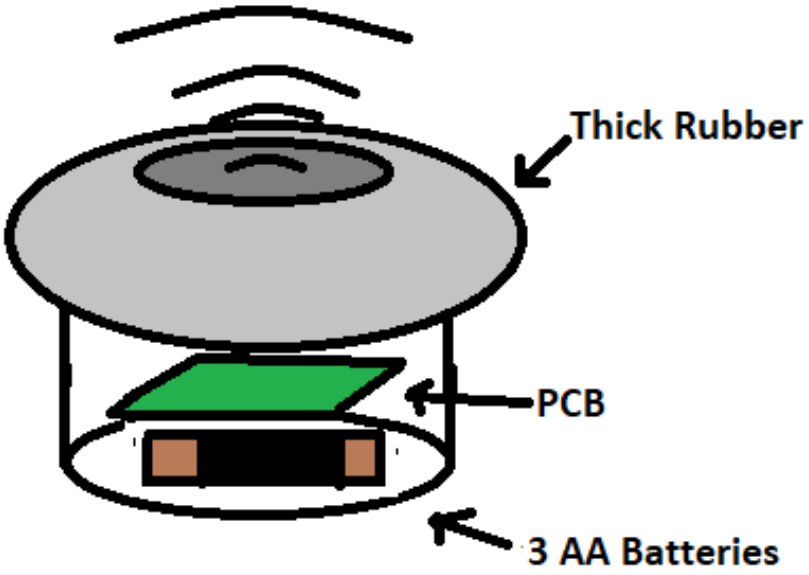


Figure 22: Sensor module housing concept, inside view.

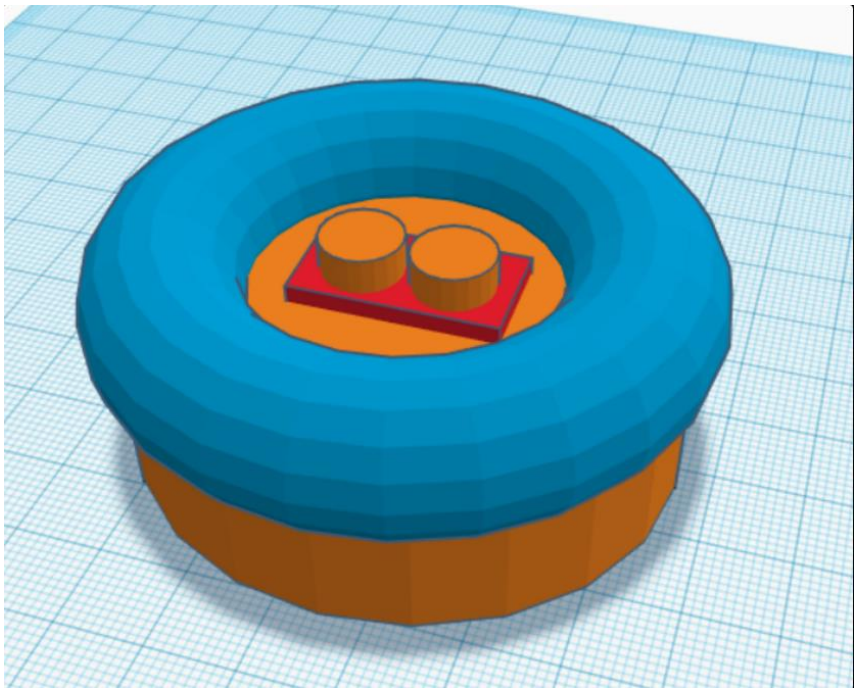


Figure 23: Sensor module housing concept, outer view.

6.2 Computer Vision System Design

This section will detail the design of the software and hardware used in the computer vision system. In this section we will give a brief overview of the system, discuss the tools used, and then detail the design and implementation of this system.

6.2.1 Overview

The system will utilize a Raspberry Pi 3 Model B+ as its central computing device. The choice of the Raspberry Pi allows for use of more complex vision detection methods such as YOLO. The use of this device also makes prototyping much easier as well allowing a more straightforward installation of computer vision related software with the ability to use a Linux based operating system. The system will utilize the ArduCam OV5647 camera as well as a 12-volt battery pack for power. With the battery pack, the system will only be able to run for a limited time, which coincides with the system constraint discussed in detail in Section 4 of this document. For a single system, the device is required to output the availability of the spots being observed. The system will also have to receive the availability of spots from the ultrasonic sensor system. After collecting the data, the device will also handle the transmission of the data to the database. The database and related web services are discussed in section 6.3 but will be referenced here.

6.2.2 Development Tools and Software

Table 30 shows what tools and software will be used to develop this system. These items were selected following the research conducted in section 3, as well as the corresponding comparison sections. Included for each tool/software is a brief description of its use within the creation of this system.

Table 30: Summary of computer vision development software.

Software	Description
PyCharm IDE	The PyCharm IDE will be utilized for all Python debugging
Python 3.6.4	Python will be used as the primary language for computer vision and transmission of data
TensorFlow and Keras	Both technologies will be used with Yad2k to implement the Tiny YOLO detection method
OpenCV 3.4.0	OpenCV will be used to implement computer vision detection methods and image manipulation
PuTTY	PuTTY is a serial communication application that will be used to communicate with the Raspberry Pi wirelessly via SSH
Git	Git will be used for all software version control
EagleCAD	Eagle will be used to create the hardware schematic for this system

BBox Label Tool	This tool will be used to annotate images for use in creation of the neural network
Ubuntu Mate 16.04	The Raspberry Pi will run a modified version of the Ubuntu Linux distro

6.2.3 Computer Vision Software Design

After the in-depth discussion and comparison of detection methods, the Tiny YOLO detection method will be used in the primary design. The HOG + linear SVM will still be tested, as the software design between both will be very similar, but only the YOLO method design will be described here.

The design will use Python in conjunction with Keras and TensorFlow. The software should run upon boot of the system and continue detecting until turned off. The software will take a picture of the of the observable space, with a pre-determined number of available spots, and return the number of cars present in the image using the YOLO detection method. The software will take a picture every 30 second and send updates to the database upon every update. The updating of the database will be handled by a Python library called pymongo. The pymongo library allows for ease of transmission, as well as updating of the data and will assist in keeping the code readable and accessible. If no change in parking availability occurs, no update to the database should take place. The Pi will send updates to the database via Wi-Fi provided by UCF.

We will show the live video feed of the parking area being observed, drawing boxes around spots that are occupied by a vehicle. This will allow testing to be achieved quickly, as any cars that enter a spot but are not detected will not be boxed. A GUI will be created to display these results. The GUI will be created using the TkInter python library, which allows for the creation of simple and functional GUIs that will satisfy our needs for this system (an example of a GUI created in TkInter can be seen in Figure 24). This GUI will be used extensively in the testing portion of our system.

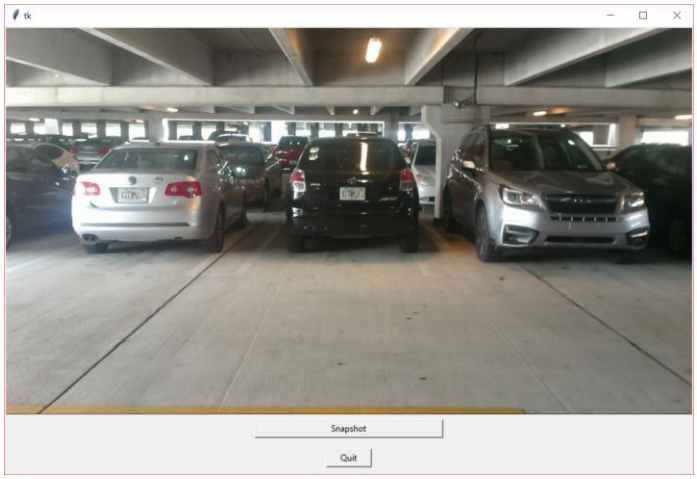


Figure 24: Example of GUI created using TkInter.

The final design should have no user interaction with the computer vision software. All detection should happen in the background without need for manual operation. Methods for testing, as well as test cases and results, will be discussed in section 7 and 8. The software flowchart for this design can be seen in section 6.2.4.

6.2.4 Software Flowchart

The flowchart for the computer vision software which will handle the detection and updating of the database.

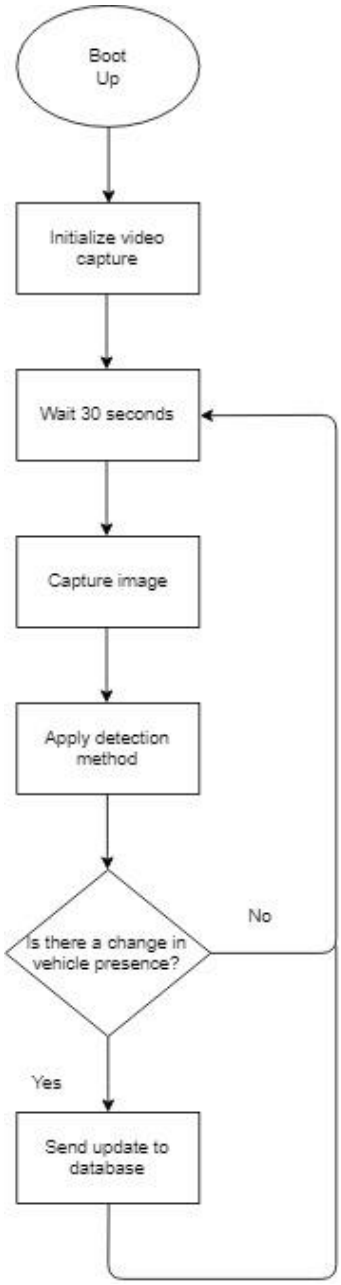


Figure 25: Software flowchart for computer vision system.

6.2.5 Dataset Creation

While there are numerous datasets already created for use with TensorFlow and Keras, the creation of our own dataset will allow the possibility of more accurate detection. The creation of our own dataset also allows for comparison with the premade datasets provided, which will benefit us in the testing phase of our project. In this section we will discuss how to create a dataset as well as the training of a neural net with the new dataset.

The first step to training any neural net is the creation of images that it can train on. A large amount of a positive and negative examples will need to be taken and processed to be fed to the training algorithm. The selected sample size for this project will be 500 total images. Each image will have to be annotated before it can be fed to the training algorithm. The BBox Label Tool will be used for the annotation process. Each individual image will have to be annotated with a box surrounding the object we want to detect, in this case a parked vehicle. An example of an annotated image using the BBox Label Tool is given in Figure 26. After we create the dataset, we will then run the training algorithm provided with Yad2K. We will continue to train the neural net until we get an average loss less than 8%. This process will take a long time depending on the GPU of the system running the algorithm and should not be run on the Raspberry Pi. After this process we can then use our dataset for use with the YOLO detection method.

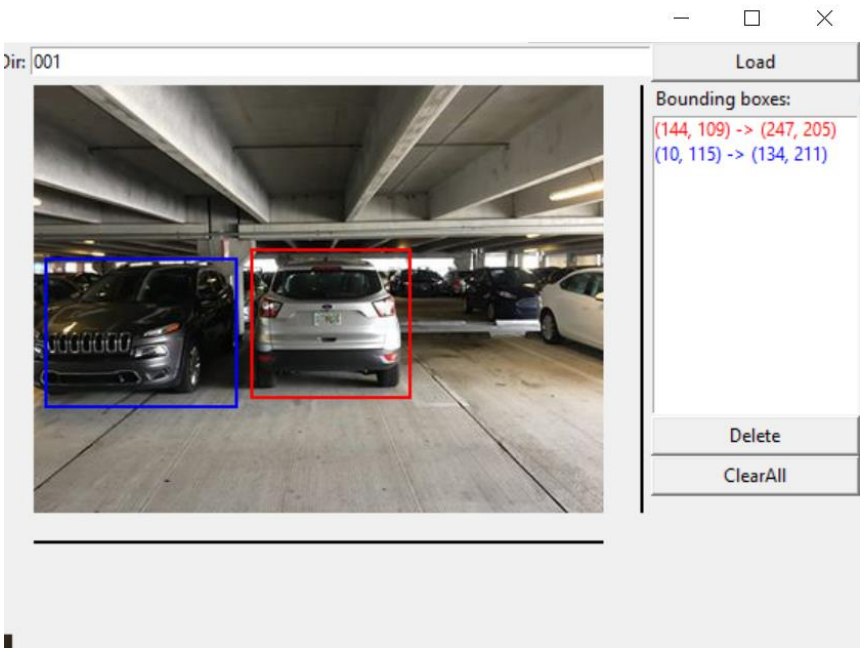


Figure 26: Example of annotated image for computer vision dataset.

6.2.6 Network Implementation

This section will detail the methods used to connect the Computer Vision sensor to the internet as well as the method used to receive data from the UltraSonic sensor system.

6.2.6.1 Bluetooth

Bluetooth will be used to communicate with the Ultrasonic system sensor. As discussed in the Ultrasonic system design, all data will be transferred to a master Ultrasonic sensor, and then be transferred to the Computer Vision sensor via their own Bluetooth chipsets. Upon receiving the data on the Computer Vision system, the sensor will take its own availability data and the data from the other sensors and transmit it to the database.

6.2.6.2 Internet Connection

The original design for the Computer Vision sensor assumed the ability to connect to UCF's WiFi network to use to transmit parking availability data to the database. In the advent of an inability to connect to a network, a second method for connection will be required. For this project, the use of a Modem LTE USB Dongle will be used to connect to a cellular network. The modem will be connected to one of the USB ports on the Raspberry Pi. The specific device used will be the Hologram NOVA. The device was created with Internet of Things (IoT) designs in mind and is perfect for use with single board computers. The service also offers the use of their services of up to 1 MB for free, which upon testing of our devices might be enough to transfer the parking availability data to the database.

6.2.7 Hardware Design

The schematic detailing the design of the computer vision camera module is provided in Figure 27.

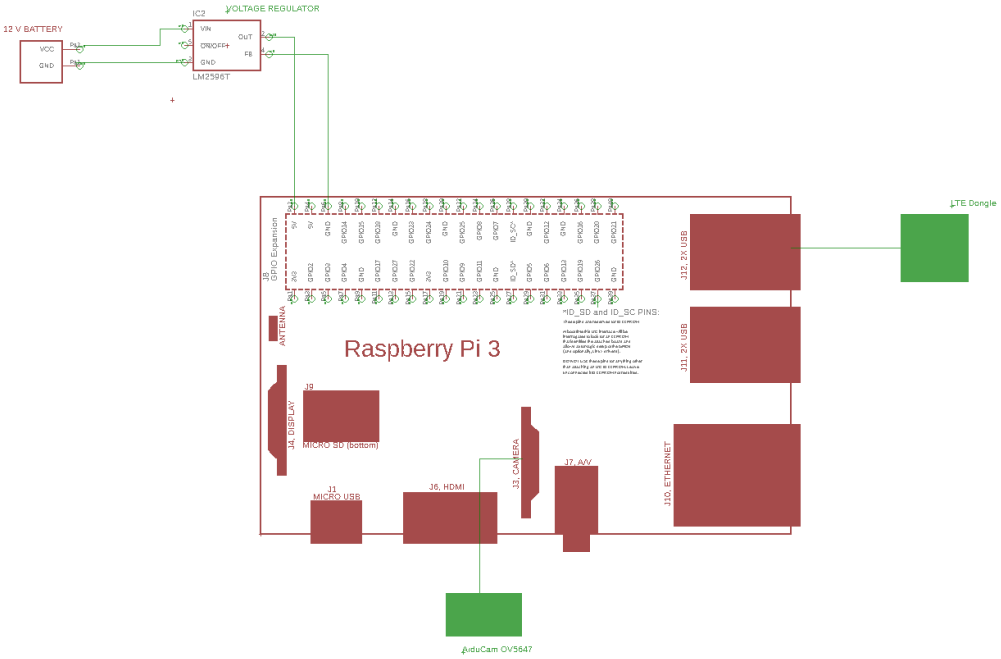


Figure 27: Hardware schematic for computer vision system.

The hardware design for this system is very straight forward. A 12-volt battery pack will used to power the system, with a DROK LM2596 voltage regulator used to step the voltage down to 5 volts. The Raspberry Pi will be powered using the GPIO pins, instead of being powered via the micro-USB port. The camera will be connected via a FPC cable using the J3 Camera connection port provided. The LTE Dongle will be connected using of the 4 USB ports provided All the wired connections between the battery, voltage regulator, and Pi GPIO, will be tested using jumper cables. However, in the final product, the wires and connections should be soldered together.

6.2.8 Computer Vision System Housing

The housing for the Computer Vision system will be described in this section. In accordance with the project specifications and requirements detailed in section 2, the dimensions of the entire system should not exceed 2.9 x 8.9 x 6.5 inches. This will ensure the housing is able to be mounted with little need for industrial hardware to support it. To coincide with the small form-factor, the expected weight of the housing should be kept to a minimum as well.

The design for the housing will be created using 3D printing. UCF provides a free 3D printing allowance that can be used at the TI innovation lab which can be used to print the housing. 3D printing the box allows a totally custom box, with the ability to define the dimensions and allow any modification required. For this design, we will need a whole created in the box for the camera to allow vision of the parking spots. The design will also require modifications for mounting purposes, as some form of external device will be needed to mount the system to the parking garage ceiling in a secure fashion. 3D printing also allows the use of lightweight materials such as nylon to create the box which will allow for an extremely lightweight product that is also incredibly sturdy.

The initial design for this box was created using Tinker Cad. Tinker Cad is a web based online CAD modelling tool used extensively in the hobbyist community, with all design shared by users allowed for use. The exact dimensions of the design are a 58 x 110 mm box with a 52.35 x 110 mm sliding box lid. These dimensions converted to inches is 2.3 x 4.3 inches and 2.1 x 4.1 inches, which fit the specifications and requirements set in the requirements section. The CAD design for the housing can be seen in Figure 28.

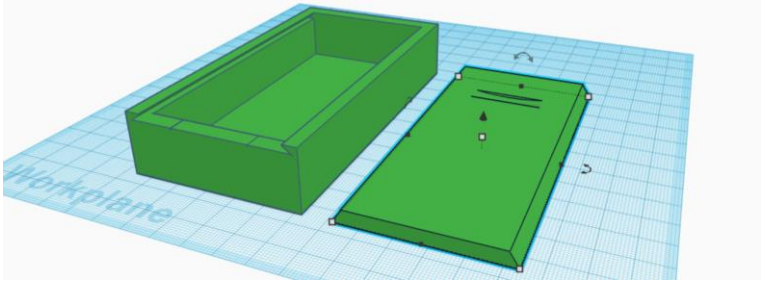


Figure 28: Housing for the computer vision sensor.

6.3 Mobile App Design

The creation of a mobile application is essential to any modern-day technology. In this section we will be covering the overview of the mobile application and the interface of the system.

6.3.1 Mobile App Overview

The mobile application will be one of two ways the users of the Park Shark system will be able to access the information being provided by the system. It will run in near parallel with the web application being created as well. The distinction between the mobile application and web application is that the mobile application will be target more towards the common user of the system. It will allow the user to be able to download and always have the nifty Park Shark application right on their phone at any notice needed. The mobile application also allows for a better user experience. Because phones do not have the same dimensions as monitors the mobile application will ensure all of the front-end will work properly and be displayed properly as well.

The mobile application will allow users to select a garage of interest and will then be provided with information on that specific garage. Each garage will hold information on availability by displaying a used to capacity ratio. This ratio will be used for a garage as a whole and for each floor in this garage. It will also have statistics on peak hours and average time to park. In the initial screen you will also be presented with a FAQ (frequently asked questions) option and a contact option. As the name implies the FAQ option will have a list of frequently asked questions with answers. The contact option will have the information to reach out to one of the developers.

When the mobile application is launch it will not require any information from the user. The user will not have to login and enter any data into the application. The application should simply display the options currently available to choose from. The user will have an option to submit an average wait time for finding a parking spot. This is only an option, and nothing is required from the user. This allows the mobile application to stay simple with the user in mind.

Although the mobile application does not have a user login it does have an admin account login. This allows the manager of the Park Shark system to edit the number of garages available on campus. The status of garages can also be adjusted. This is useful in the case of a garage being under construction or expanded. A perfect example is garage C at UCF which was temporarily unavailable to latter become available with many more parking spots. There are also special events that may require a garage to be closed off or made partially available.

6.3.2 System Interface

This user interface holds all the GUI (graphical user interface) components. This component ensures that all of the front-end modules are able to communicate with one another as well with the networking component when needed. React Native

will be used to create the user interface and is also used to interface the front-end with the networking by using the Fetch API. Further system interface can be reference in section 6.4.3.

6.3.3 Mobile App Block Diagram

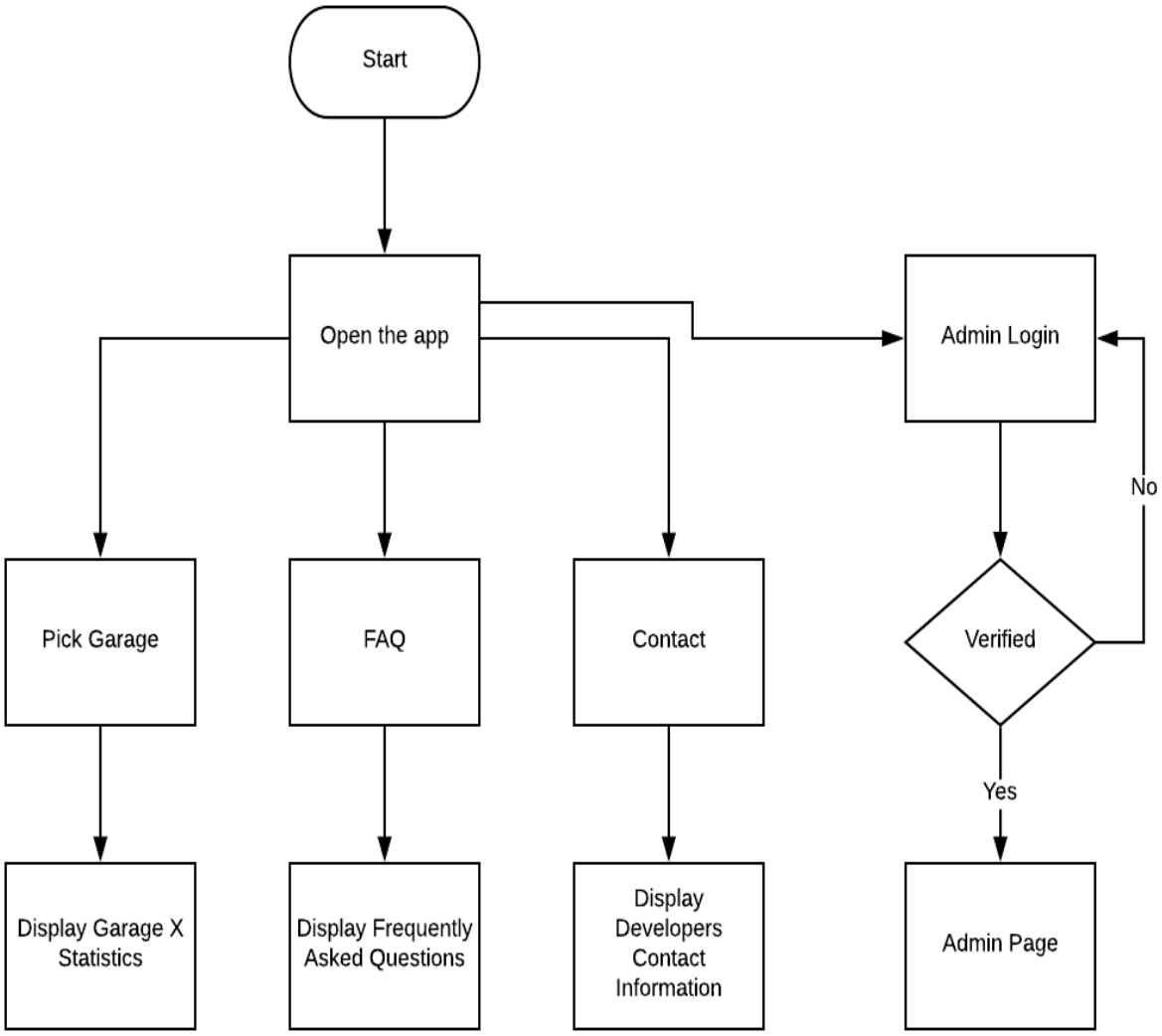


Figure 29: Block diagram for mobile application.

6.3.4 User Interface

This section covers how the mobile application looks like at its early stage. This is not a comprehensive or example of the final product. The mobile application should be able to handle both the portrait and landscape mode. Depicted below in Figure

30 and Figure 31is the mobile applications home screen in portrait and landscape mode.

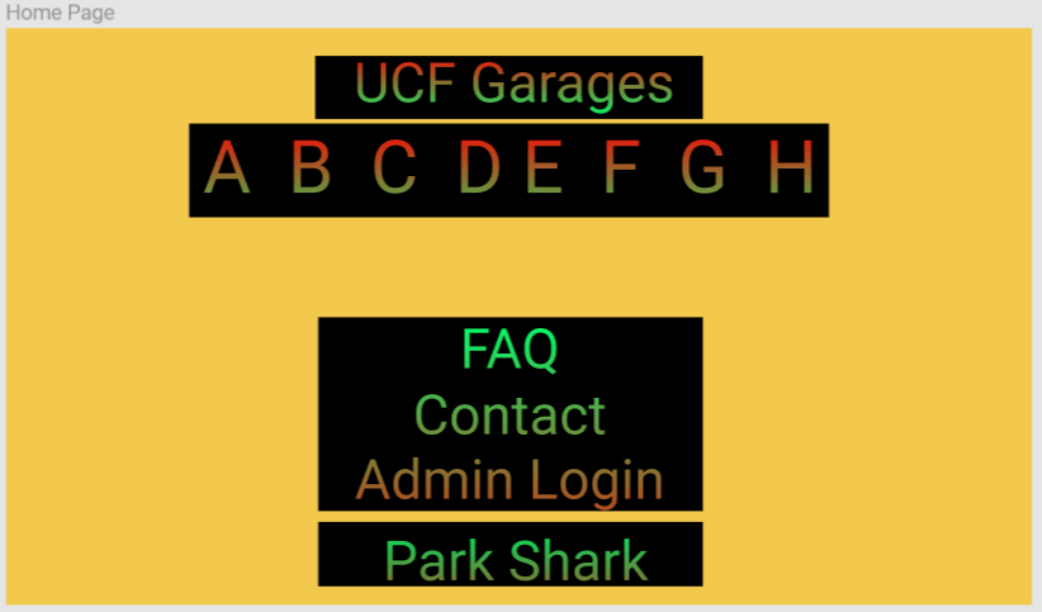


Figure 30: Home Screen Interface in Landscape Mode.

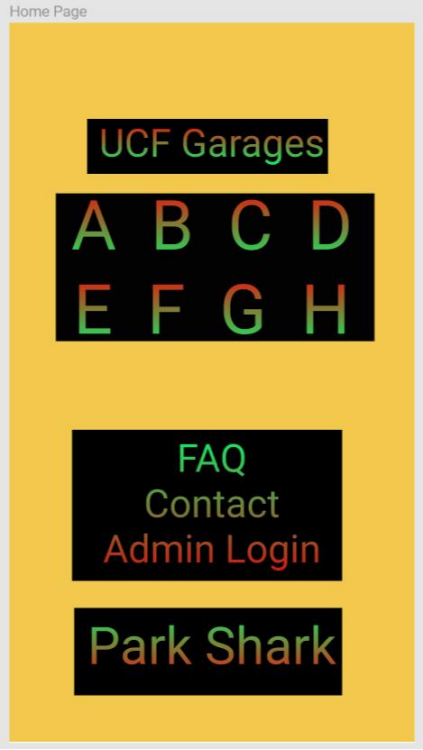


Figure 31: Home Screen Interface in Portrait Mode.

Once the user reaches the home screen he will be presented with the option to choose from a selection of garages. There will also be a FAQ, Contact, and Admin Login tab at the bottom of the screen. An example of the screen after a garage is selected is displayed bellow on Figure 32.

Figure 32: Selected Garage Availability & Statistics.

Lastly, the Contact and FAQ screen is not provided because it is still in question how and if these components will still be displayed. The Admin Login screen is a must for the mobile application because the manager of the Park Shark system should be able to modify the garages availability without the use of a computer. The Admin Login Screen and Admin Screen are both displayed on Figure 33.

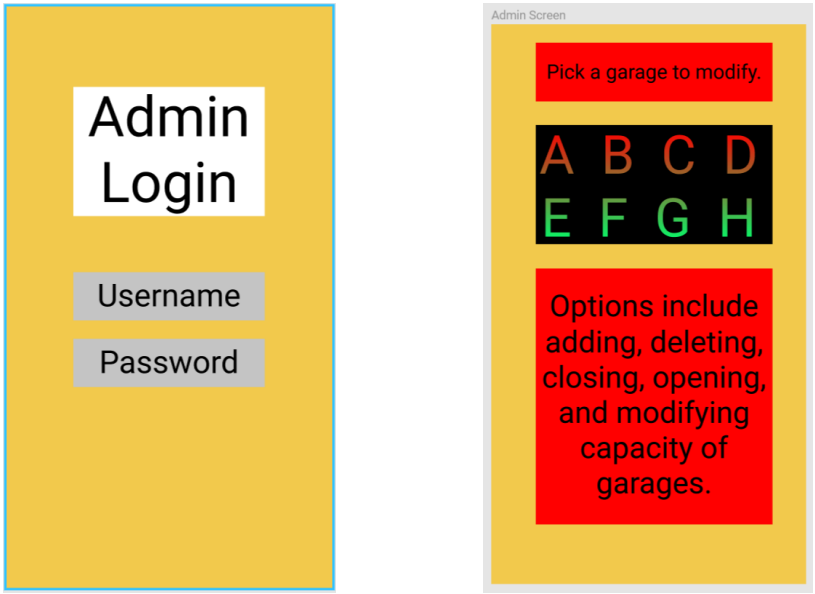


Figure 33: Admin Login Screen.

6.4 Website Design

This section details the design of the website. This website will be the secondary means of viewing the parking information status, where the mobile app is the primary interface. However, the browser website will have more functionality for changing the settings and overall administration of the parking system. From the perspective of a user looking for parking however, both versions will provide similar functionality.

6.4.1 Website Design Overview

The website for Park Shark will be one of the main ways for users to access the data provided by our system. The website will be a comprehensive multi-user application that will allow all users the ability to access parking information. Users will be able to select a garage and be provided with the current capacity of the garage as well as related statistics such as peak hours and average time to park. The website will also feature a brief about page detailing the functionality of our system as well as links to view the code for the project. The site will also feature a contact page that will provide links to all developers. This system will be implemented as a web app to allow all users access across all platforms. While this site will be developed for desktop as well as mobile browser use, the mobile app detailed in section 6.3 will provide a more complete experience for mobile devices. All data collected from the sensors will be stored in a secure database, which will be provided upon query from the user. The same database will be used with both the website and the mobile interface.

The user will not be required to have an account or enter any login information, all data should be provided to any user of the site. The user will also have the ability to submit parking wait time. On the parking availability page, the user can enter how long it took them to find a spot which will then average with all other times and reflect on the info page. The user will also be able to send a message to the developer to report any bugs or misinformation. This feature will be present on the contact page and will send the message to all associated developers.

The system will feature an admin account login page. Through this account the admin will be able to manually add and remove garages, as well as disabling garages. A garage being disabled signifies that the garage is closed and is not available for parking. This feature can be useful in the event of maintenance or special events. Specific spots within a garage or lot will also be able to be disabled, in case only certain spots are reserved for events rather than garages. These changes will also affect the displayed available parking spots by not considering those that are closed or reserved for events.

6.4.2 Development Tools

Following the research and comparison in section 3.9, the following table details the selected technologies for the website frontend. Included for each tool/software is a brief description of its use within the creation of this system.

Table 31: Summary of website development technologies.

Tool/Software	Description
Atom	Atom will be used as the main development environment
Git	Git will be used for all software version control
Figma	Figma will be used to design all UI interfaces
Postman	Postman will be used to simulate calls to the database and will assist greatly in the design and testing phase of the website
Robomongo	Robomongo will be used to manage the MongoDB database with the ability to add and delete items from the database manually
React	Will be used to create the front-end of the application
Redux	Will be used in tandem with React to handle all transfer of data from the back-end to the front-end
Node.js	Will be used to create the back-end of the application
MongoDB	Will be used to create the database for this application
Mongoose	Will be used in tandem with MongoDB and Node.js to help communicate between the two

6.4.3 Website System-interface Design

This section will detail the different components of the system, as well as which technologies will be used for each. A more in depth look into to the technologies, as well as the decision to use certain technologies, can be found in section 3.9. The complete design can be broken down into three components:

- User Interface
- Network
- Database

The full architecture of these three components can be viewed in Figure 34.

User Interface:

This component will contain modules that will handle the front-end of the application. All modules within this component must communicate with each other as well as with the modules in the Networking component. React will be used to render all changes to the application view. Redux will be used to help define the application state, gathering the information from the back-end to give to React to update the view. All of these technologies are used in tandem in order to ensure the displayed information is always as up to date as possible without the end user having to refresh the page to obtain a more current status.

Network:

This component will contain modules that will handle the back-end of the application. All modules within this component must communicate with each other as well as components in the User Interface as well as components in the Database. NodeJS will handle all of the back-end of the application. Mongoose will be used to assist NodeJS with communication with the MongoDB database.

Database:

This component will contain modules that will handle the creation of the database for the application. MongoDB will handle all the storage of the parking availability. There will be two models for the database, the Admin model and the Garage model. The Admin model will store all relevant account information for an admin account. The garage model will store all parking information, such as spots available on each floor, as well as store the related statistics such as peak hours and average park time. There will me one Garage model for each parking garage at UCF.

6.4.4 UML Diagrams

This section will provide several UML diagrams used to implement the website design. These diagrams provide further detail into the design of the website.

High Level Class Diagram:

The Class diagram, seen in Figure 35, shows the relationships between the different pages, the related items showed on each, as well as any associated functions. The navigation bar is present on each page and serves as the main tool to navigate the website. The functions available to the user can be seen in the viewParking class as well as the contact class. The functions available to the admin can be seen in the adminPage class.

Use Case Diagram:

The use case diagram shows what pages and functions are available to the user and the admin. The diagram can be seen in Figure 36.

Network Diagram:

The network diagram shows the basic design for the database implemented with MongoDB. The diagram showcases the required get, post, patch, and delete functionality that is required for our user and admin functions. The diagram can be seen in Figure 37.

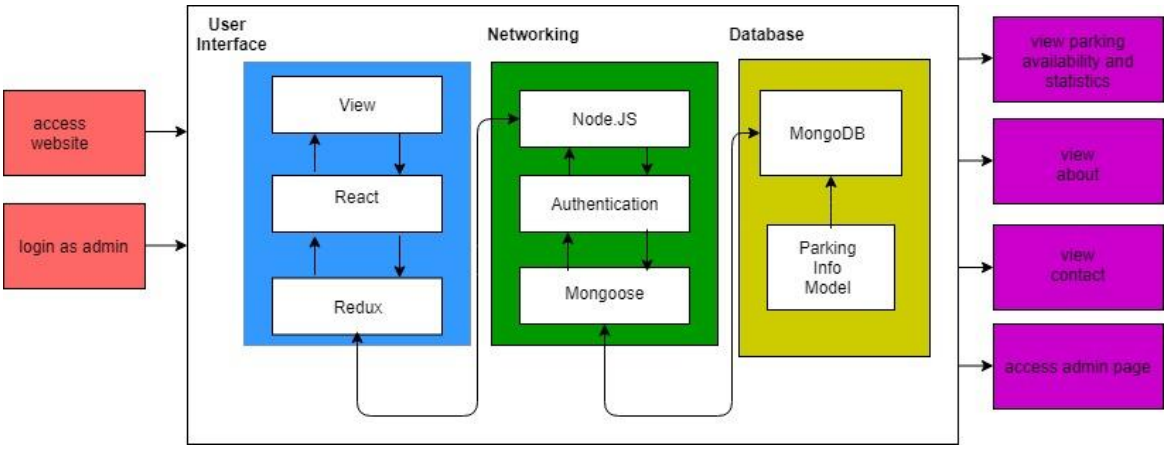


Figure 34: System-interface architecture.

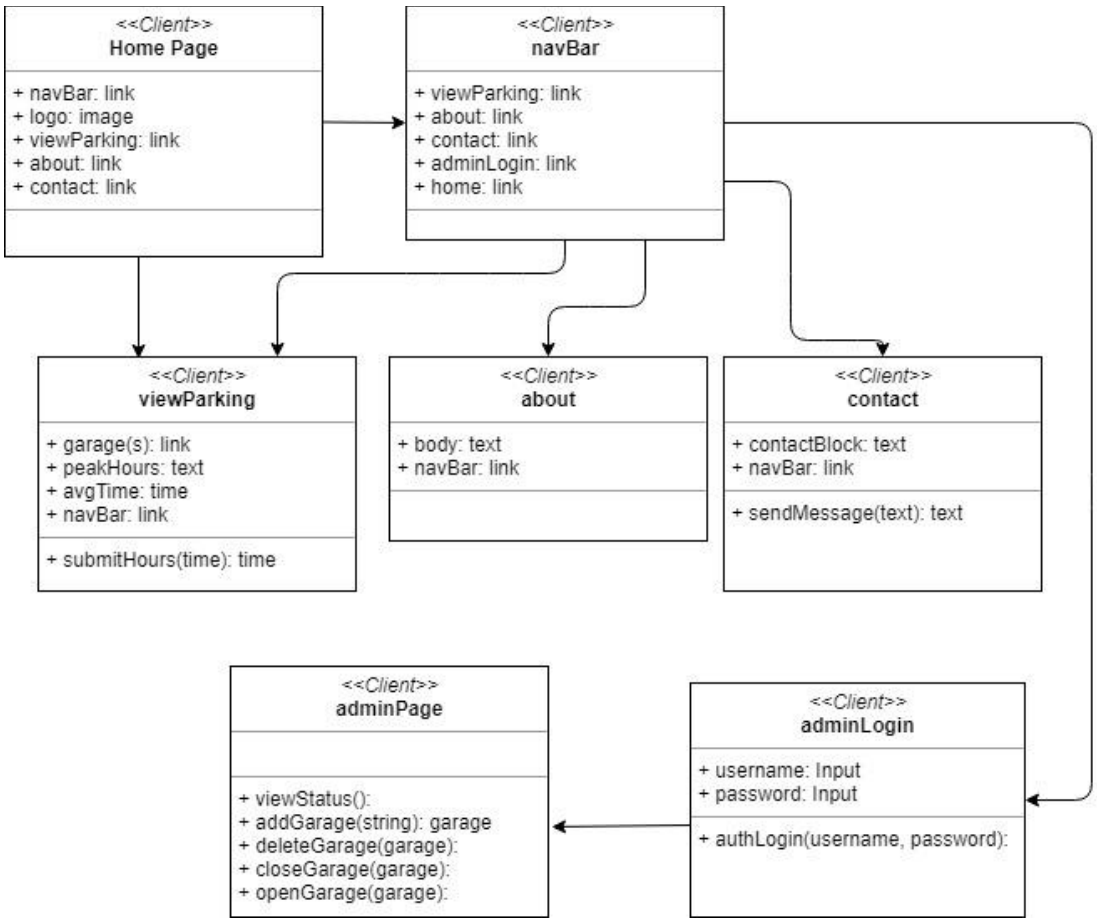


Figure 35: High-level class diagram.

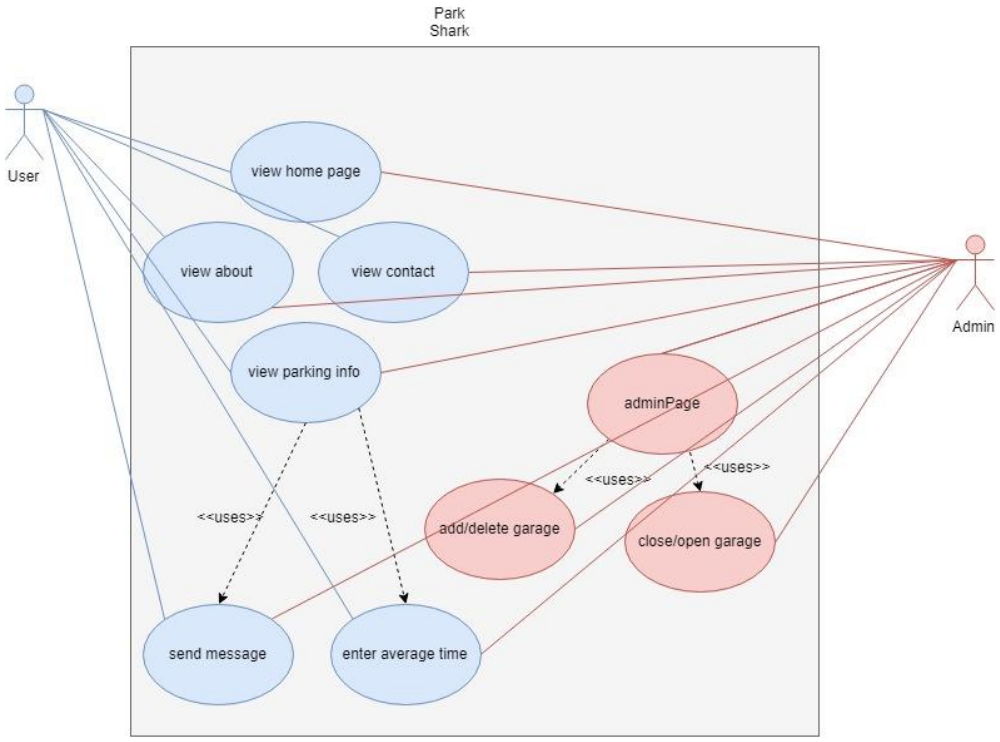


Figure 36: Use case diagram.

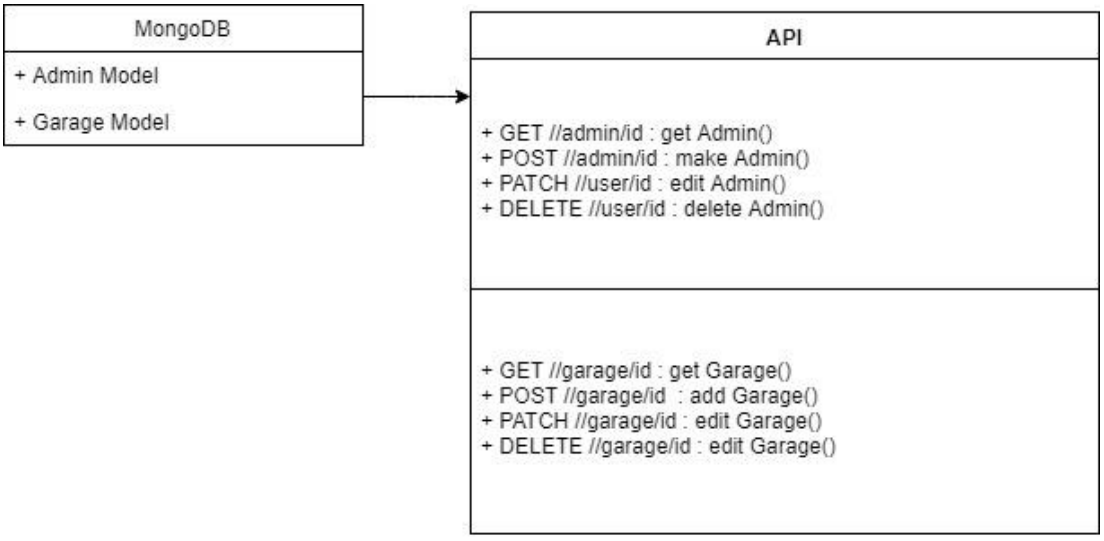


Figure 37: Network design diagram.

6.4.5 User Interface Design

The user interface was designed using Figma. The site features a total of 6 pages. The site was designed with simplicity in mind. The images shown below reflect the initial design of the User Interface. The final design may differ based off design changes and actual implementation.

Parking Info Screen:

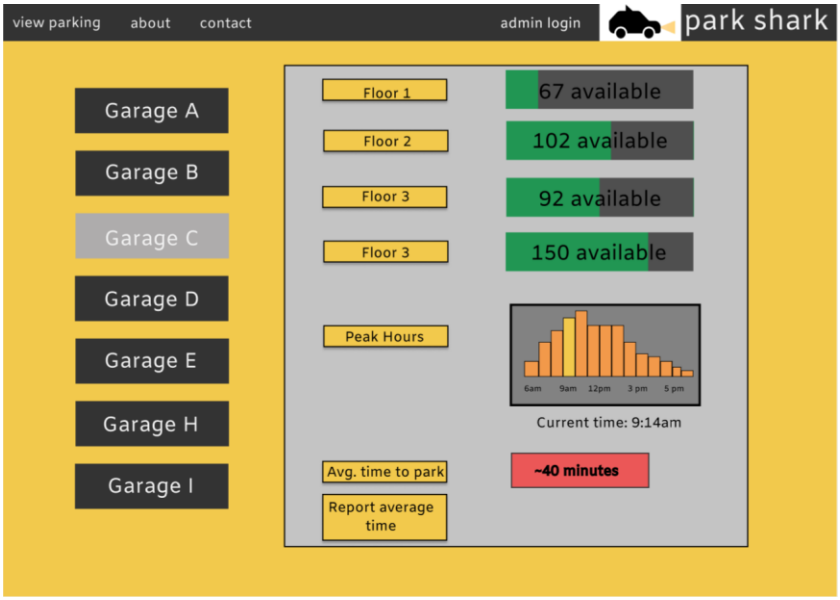


Figure 38: Initial parking information page design

Contact Page:

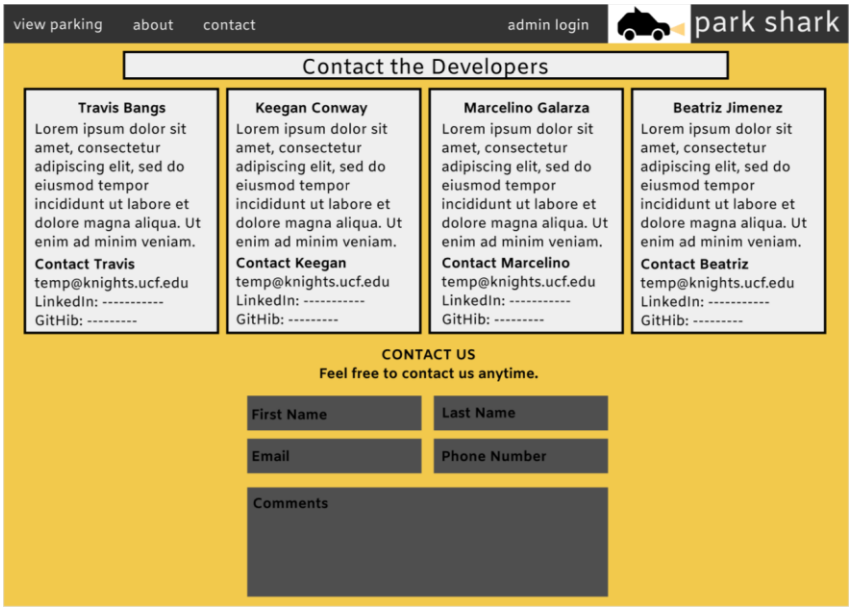


Figure 39: Initial contact page design.

Admin Login and Admin Page:

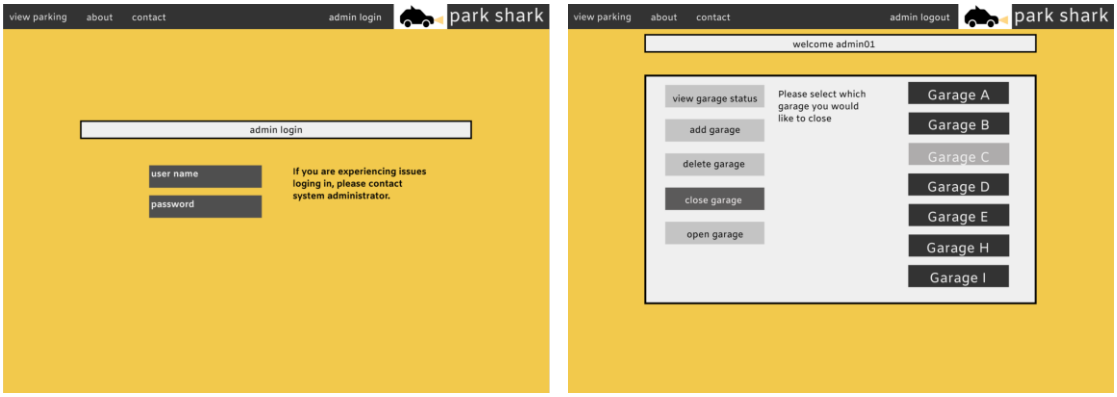


Figure 40: Administrator login and controls page design.

About Page:

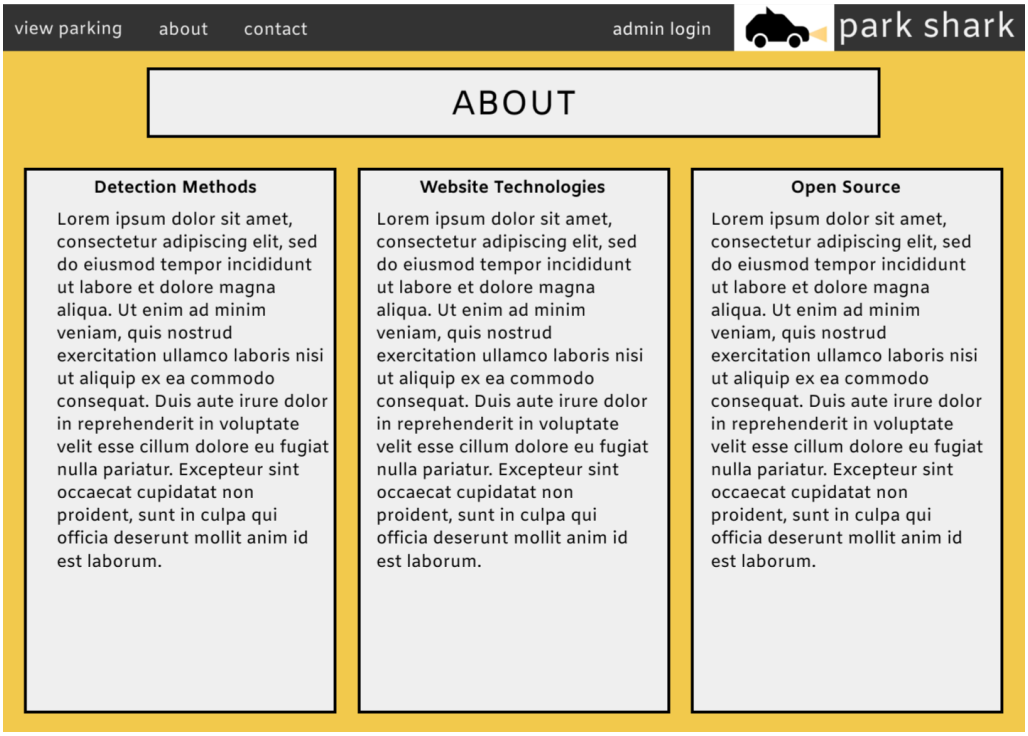


Figure 41: About page design.

6.5 Design Summary

The above sections represent the initial design for the Park Shark system. The design represents a rough design that has not been prototyped to an acceptable level. The design may change dramatically from its current form to the final product created in Senior Design 2. This section will provide a summary of the design sections for the Ultrasonic sensor, the Computer Vision Sensor, and the Website and Mobile Application.

The wireless ultrasonic sensor will be powered by 3 AA batteries, and will dynamically change its polling rate depending on the current occupancy percentage of the area or garage it is configured to be a part of. The sensor will communicate via Bluetooth with the computer vision sensor to provide the status on parking spaces to the central parking system server. The battery life should last at least 1 year before the batteries will need to be replaced. Sensor can be mounted on the floor, wall, or ceiling near a parking spot.

The Computer Vision sensor was created to monitor multiple spots at once utilizing CV methods. The hardware for the sensor utilizes a Raspberry Pi for all object detection and communication. The device is powered by a small 12-volt battery pack and uses a voltage regulator to step down the voltage to 5 volts to power the Pi. The device will be housed in a small box and mounted to the ceiling of the garage. The minimum number of spots required to provide information on is three. The device utilizes Keras and TensorFlow to handle all the detection methods, and the Mini YOLO detection method will be implemented with both. The Mini YOLO method will be used for all object recognition purposes and will utilize a custom dataset created specifically for this project. The use of 600 or more image will be used to create this dataset, with each individual image being labeled by hand.

The website and mobile app for this project will be used to convey all information gained from the Park Shark sensor system. The website front-end will be created using React library while the mobile application will utilize the React Native library, and both will utilize Express to assist with communication with the backend. The backend for both will utilize a NodeJS backend with MongoDB serving as the primary database for this system. Both applications will have similar functionality. Students will be able to use the website and app to access parking availability information for any garage in the system, as well as have the function to report average parking times on the info screen. An admin of the application will be able to login to an admin account and be able to add, delete, close, and open garages.

The overall system described should accomplish all requirements and specifications detailed throughout this report. The overall goal of this system should be to relay all parking availability information in a concise and efficient way, where the user feels that are able to incredibly accurate information in a concise and efficient way. While modifications may be made to this system throughout the implementation stage, this goal will remain as the primary end goal for the team.

7.0 Prototyping

With all the components chosen in in Section 6.0, the Design section, a schematic prototype can be made. This schematic can first be constructed using the nRF52832 Dev Kit and breadboard, then once functionality is confirmed, it will be built on its own separate PCB.

7.1 Circuit Board Schematics

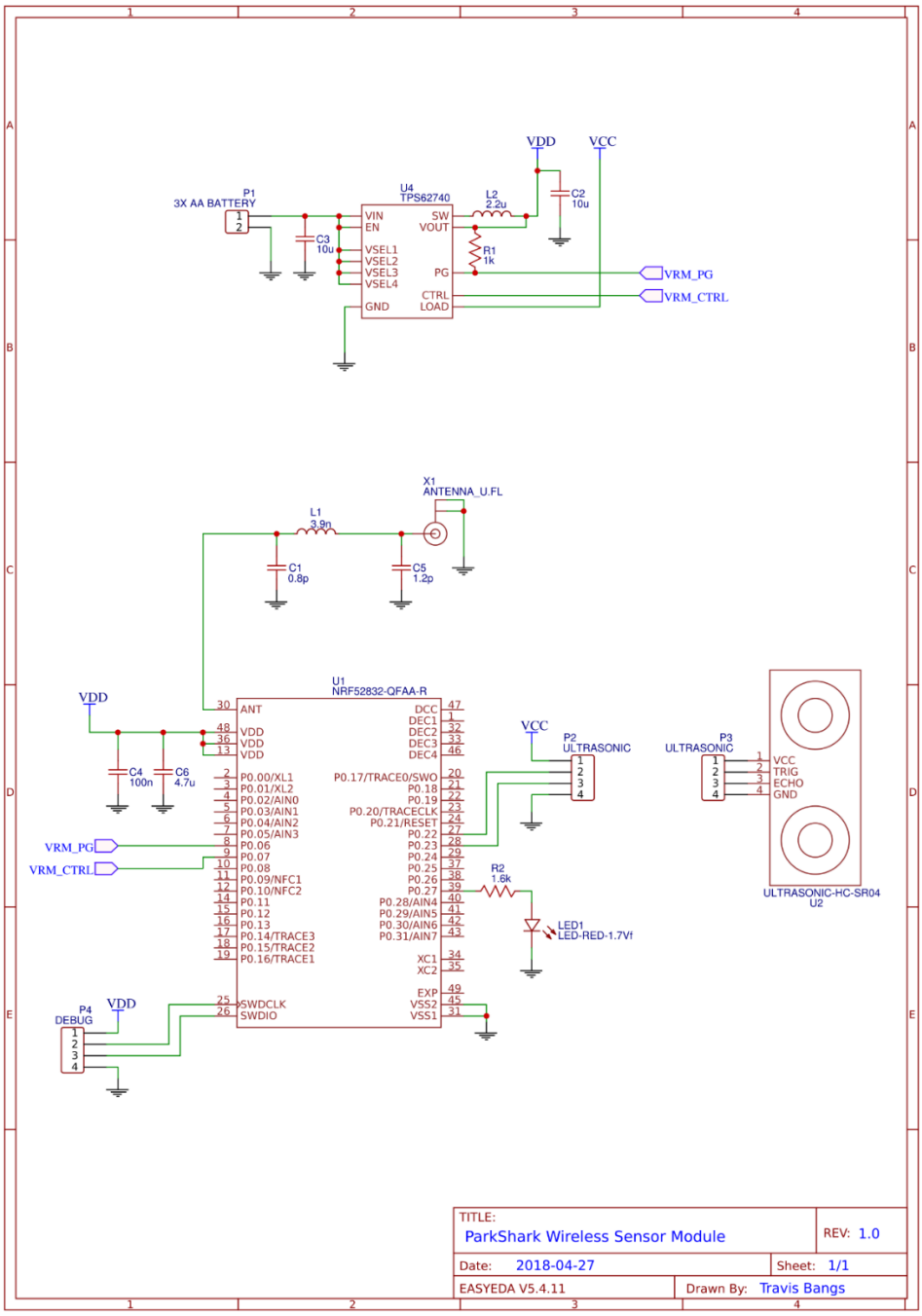


Figure 42: Circuit schematic of wireless sensor module.

The circuit schematic, presented in Figure 42, is divided into two sections. The top section constitutes of the power supply, made up of the battery connected to port P1, and the voltage regulators. This power supply has two outputs: a constant 3.3V VDD output, and a secondary switched 3.3V VCC output, that will be turned on an off by the nRF52 microcontroller via the VRM_CTRL pin. VRM_PG is a “Power Good” status pin indicating that the voltage regulator is correctly outputting 3.3V. VRM_PG will turn Low when the output voltage starts dropping below 3.3V, indicating that the battery is getting low and should be replaced. VDD is used to power the microcontroller and Bluetooth module, while VCC is used to power the ultrasonic sensor. There is a 4-pin debug port P4 that will be used for uploading firmware to the nRF52 microcontroller, as well as debug purposes. This port is connected to a 4-pin debug port on the nRF52 dev kit. Port P2 connects to the 4-pin header on the ultrasonic sensor. The schematic was created using the EasyEDA online schematic creation software.

7.2 Preliminary Board Layout

EasyEDA also allows for converting the schematic file into a board layout. The board has a copper ground plane on the back layer, which was recommended by the nRF52 reference circuits. Also suggested was the relative placement of the inductors and capacitors for the antenna track to ensure maximum possible RF performance and avoid interference. To help achieve this, the antenna plug has been placed as far away as possible from the power supply hardware, which contains the only other oscillating signal on the circuit. The board layout is presented in Figure 43.

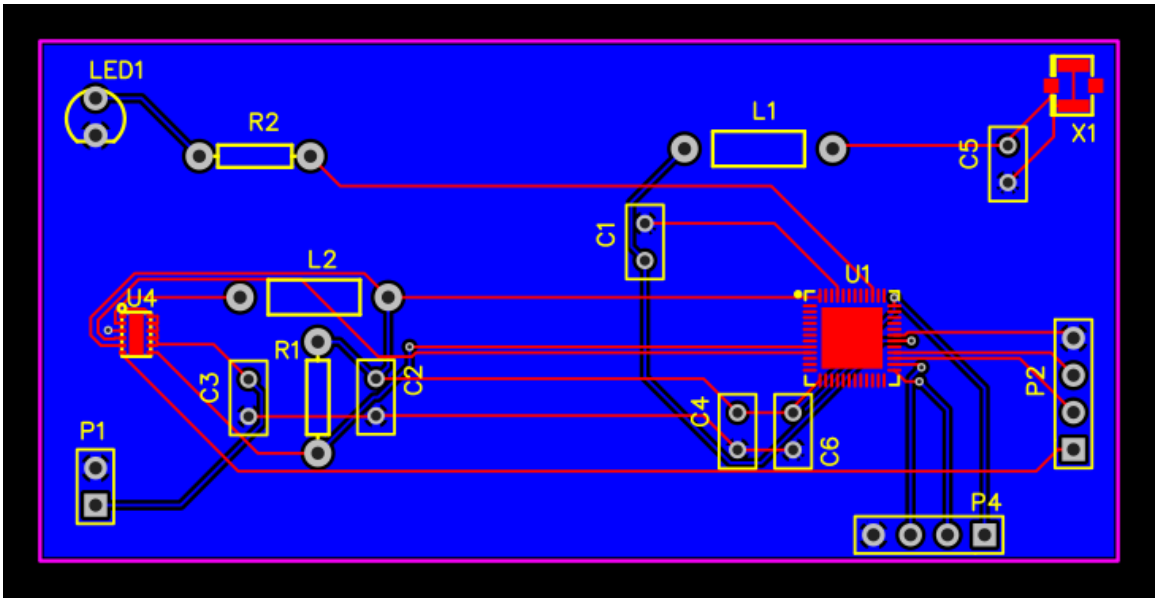


Figure 43: Board layout of wireless sensor module

The board layout is only preliminary; it is most likely that the layout will have to be carefully adjusted during the build portion of the project, due to the necessity of tuning the hardware to maximize the performance of the RF antenna.

7.3 Bill of Materials

The bill of materials for the circuit board is included in Table 32.

Table 32: Bill of materials for wireless sensor module circuit.

Item	Value	Qtx	Price	Package	Mfr	Manufacturer Part
U1	NRF52832-QFAA-R	1	$3.5143	QFN-48_6X6X04P	NORDIC	NRF52832-QFAA-R
U2	ULTRASONIC-HC-SR04+	1	$1.4400	HC-SR04+		
L1	3.9n	1	$0.0029	AXIAL-0.4	FH	VHF060303H3N9ST
C1	0.8p	1	$0.0187	RAD-0.1	MuRata	GJM1555C1HR80BB01D
C4	100n	1	$0.0147	RAD-0.1	AVX	06035G104ZAT2A
P2,P3	ULTRASONIC CONN	1	$0.0244	HDR-4X1/2.54	Ckmtw	210S-1*4P L=11.6MM
C5	1.2p	1	$0.0090	RAD-0.1	FH	0603CG1R2C500NT
C6	4.7u	1	$0.0141	RAD-0.1	FH	0603B475K100NT
L2	2.2u	1	$0.0266	AXIAL-0.4	FH	2.2UF 16V 10%
C2,C3	10u	2	$0.0085	RAD-0.1	ValuePro	10uF 16V 5*11
R1	1k	1	$0.0022	AXIAL-0.3	UniOhm	1210W3F1001T5E
X1	ANTENNA_U.FL	1	$0.0158	U.FL	HRS	U.FL-R-SMT-1(80)
P1	3X AA BATTERY CONN	1	$0.2364	HDR-1X2/2.54	TE Conn	826629-2
LED1	LED-RED-1.7Vf	1	$0.0115	LED-3MM/2.54	KENTO	ϕ3 Red
R2	1.6k	1	$0.0034	AXIAL-0.3	KOA	RK73B1ETTP162J
U4	TPS62740	1	$1.9838	R-PWSON-12	TI	TPS62740DSSR
P4	DEBUG	1	$0.0244	HDR-4X1/2.54	Ckmtw	210S-1*4P L=11.6MM
		Total:	$7.3591			

The total price per unit of the circuit board component of the sensor modules, including the ultrasonic sensor, is $7.36. This price does not include the cost of the PCB manufacturing or the physical enclosure. All of the parts except for the HC-SR04+ ultrasonic sensor can be supplied by LCSC, while the HC-SR04+ must be sourced from an alternative supplier. A supply of HC-SR04+ has been found on Fasttech.com.

8.0 Testing

Before the final products are assembled, each element of the project must be tested for correct operation before being put into the system. This will ensure that none of the components used in the product are faulty and operation will be correct. The testing procedures are split into two sections: hardware testing for the wireless sensor module, and software testing for the computer vision module. All parts of both elements must have proper operation for a completely functional parking monitoring system.

8.1 Hardware Testing

The hardware testing section details the testing procedure to verify the components of the wireless ultrasonic sensor module. Each component will have its own procedure for verifying correct function.

8.1.1 Testing Overview

Each of the major components (voltage regulator, nRF52 MCU, ultrasonic sensor) will be tested before being placed on the circuit board, in order to ensure none of the parts are faulty before the circuit board is constructed. This will save effort in manufacturing as faulty components can be replaced with working ones before assembly onto the board.

8.1.2 Microcontroller Testing

To test the proper operation of the nRF52 microcontroller/Bluetooth transceiver, an alternate programming flash file will be created. Instead of checking the ultrasonic sensor, this test program for the nRF52 will simply flash on and off the LED on the circuit by turning on and off GPIO P.027. If the LED flashes as expected, it will be confirmed that the nRF52 was fully capable of: accepting power input, proper function of sleep modes, having software written to the device, and controlling of the GPIO output. The microcontroller will be supplied with a 3.3V input, just as it is supplied in the final circuit. However, for testing purposes, the microcontroller can be powered directly without a voltage regulator, by using a DC power supply available in the UCF Senior Design lab, which will help to rule out the voltage regulator from being the cause of the chip not powering up, if such a case occurred.

Once the nRF52 has been tested with the LED flash program, it will be flashed with another sample program that will open a simple connection to a Bluetooth receiver on a computer. For this, the antenna circuit will have to be connected. Then, the message “Hello World” will be sent to the receiving computer at various transmitting (Tx) power settings. Tx at 0dB will be tested first, then it will be tested again at the 4dB Tx setting. The nRF52 chip itself is not capable of sending a higher dB of Tx power; an external power amplifier would have to accomplish this. However, the wireless sensor module will not be making use of any Tx power higher than 4dB, so this is not a concern for testing purposes.

Once the software flashing, pin input/output, and wireless connectivity features are confirmed to be working for a particular nRF52, it will be soldered onto the circuit board to be placed into the final product.

8.1.3 Sensor Testing

Before creating an actual sensor module, the sensor's range of detection must be tested. The sensor claims to be able to detect objects from 2cm up to 4 m. To implement the ultrasonic sensors in our design, the sensor must be able to detect vehicles up to 10 feet away because the height of the ceiling in parking garages can vary. The first test will involve setting off the ultrasonic sensor to detect a solid, smooth surface object a 1 foot away. The distance of the object will be increased by an increment of 1 foot each time while keeping track of the distant by recording the measurements made with the measuring tape. This way, the maximum detection range of the sensor can be measured.

Another test the sensor will undergo is seeing how uneven surfaces affect the sensor's readings. If the sensor is mounted on the wall, the ultrasonic waves transmitted by the sensor would interact the front side of vehicle which is technically an uneven surface. According to research, uneven surfaces can affect the sensor's reading because not all the ultrasonic waves that hit the uneven surface are reflected to the sensor. Therefore, the sensor will go through a series of trials to see if the sensor is able to detect the front side of a variety of vehicles.

An additional test the team shall perform on the ultrasonic sensor is measuring the time it takes the sensor to detect a vehicle, in other words, its response time. The Park Shark system wants to provide accurate and timely parking statuses that users can trust and rely upon. To do so, the sensor must be able to respond quickly so the information can be ready to be sent to the database as soon as possible.

8.1.4 Voltage Regulator Testing

The final component to be tested before installation into the printed circuit board is the voltage regulator, the TI TPS62740. Because of the chip's extremely small size, a separate breakout board will be created, which will connect the chip's small pins to regular sized pin headers for easy testing on a breadboard. Then, all of the components in the Power Supply section of the circuit schematic (Section 7.1) will be attached to the chip on a breadboard, as all of these are necessary for the chip's proper function.

Using a DC power supply available in the UCF Senior Design lab, a 5 volt input will be supplied to the power supply circuit's two battery input terminals. Then, using a multimeter, the VDD output will be tested to ensure that the voltage regulator correctly outputs 3.3 volts. Once the VDD output is verified, a second channel of DC input will supply 3.3 volts to the CTRL pin of the voltage regulator chip. This will enable the VCC output, and VCC will also be measured with respect to ground using a multimeter. Then, the 5V DC input will be removed from the battery terminals, and a pack of 3 AA batteries will be attached to the battery header. If both VCC and VDD read as 3.3 volts in both cases, then the voltage

regulator should be considered as operational, and the VCC signal should be controllable by an external circuit such as the nRF52's microcontroller.

8.2 Software Testing

Each component of our web application and mobile application should be tested to verify and validate that they meet the stated requirements. Each test procedure is described in detail for both the web application and mobile application.

8.2.1 Web Application Testing

There are six main components we are testing for: functionality, usability, interface, compatibility, performance, and security.

8.2.1.1 Functionality Testing

The functionality test is used to check if the software is giving the same output as required by the end-used. The first thing that is being check is links. Every link that is on the web application should be tested to ensure it goes to specific location. Next it will test on forms, which are the components that allow a user to input data which is then sent to a server. Check for wrong inputs, which can be check from a list of restrictions on user input for each form. Lastly it is checking that the database is working properly by checking if all database queries are executed correctly. Also, if data is being received and updated properly. Table 33 is a checklist for functionality testing.

Table 33: Functionality testing checklist.

Testing	Components
Links	Internal Links, External Links, and Broken Links.
Forms	Field validation, wrong input, optional and mandatory fields.
Database	Receiving and updating data.

8.2.1.2 Usability Testing

The usability testing is to ensure that all components of the web application are user friendly. The interaction between the user and web application will be measured in this testing. Things that are being check in this test are navigation, appearance, content, and ease of learning. First, we want to ensure that the user is able to navigate through the web page. Being able to scroll in any direction if needed. As well as being able to access all other components of the web application. Although the appearance may seem insignificant to some it is very important to some users. Check to see if all front-end components are properly aligned and spaced. There should be a consistency of look through all pages. It should also be easily visible by the user by using appropriate colors. The content on the web application should be checked for grammatical errors, font style with color, and meaningful content. Lastly is checking for ease on learning to use the web application. This is being tested by checking to see if all other components

are tested within themselves and with each other. There should also be some user help information.

8.2.1.3 Interface Testing

The interface testing is used to check if all interactions with the web application are working properly. The server should be able to communicate with the web application. The database and web application interface should be working properly. Check if interactions between servers are working, and if not ensure an error message is displayed to the user. User interrupts should also be tested to ensure the web application acts accordingly in these situations.

8.2.1.4 Compatibility Testing

The compatibility testing is checking to see if the web application works with different kinds of users. The web application should be tested on different variations of browsers, operating systems, and mobile devices. Each browser has its own way of doing things. They have their own configurations and settings and our web application should be able to work in these conditions. There may be issues with operating system compatibility because not all APIs are available to all operating systems. The web application is capable of mobile browsing, so it will be tested on multiple devices with variations of browsers as well. The checklist is displayed in Table 34.

Table 34: Compatability testing checklist.

Testing	Components
Browsers	Google Chrome, Firefox, Safari, Opera, Konqueror, Internet Explorer.
Operating Systems	Windows, Linux, Unix, MAC.
Mobile Devices	Apple iOS, Android, Windows Phone, Tablets.

8.2.1.5 Performance Testing

The performance test will check the web applications capability on how much can be loaded and how much stress it can take. This is very important because with UCF being one of the largest universities in the country we need to ensure the web application can handle a large amount of traffic. For load testing multiple users will be requesting for the same page. Check to see if the web application can sustain a large amount of request and large amounts of input data. Simultaneous connections to the web application should also be tested. The stress testing is to see how much the web application will be able to handle until it breaks. Check to see how the web application reacts to heavy stress and how it recovers from it too. Table 35 is a checklist for performance testing.

Table 35: Performance testing checklist.

Testing	Components
Load	1. System behavior at peak loads. 2. Number of users per time. 3. Large data accessed.
Stress	1. Continues load. 2. Performance of memory and CPU.

8.2.1.6 Security Testing

The security testing is used to detect potential vulnerabilities to the web application. The main concern is authentication for the administrative login. Using internal administrative URLs should not work without logging in first. Validation of username and password should be check. Passwords should not be easy to acquire so restrictions on passwords are in place. Test to see if passwords meet the requirements for minimum length and special characters. Check the web applications response to these three variations of attempted logins wrong username with right password, right username with wrong password, and wrong username with wrong password. These should all reject the user from logging into an administrative page. Ensure that a user can connect to these pages when the correct username and password are entered.

8.2.2 Mobile Application Testing

The mobile application testing is going to cover seven key areas that could fail to meet the systems requirements. They are usability, compatibility, integration, memory, performance, installation, and security.

8.2.2.1 Usability Testing

The usability testing checks to see how user friendly the mobile application is. It focuses on the ease of use, ease to familiarize with the system, and satisfaction with entire experience. The usability testing for the mobile application will start as soon as our design is being implemented. It will also be tested once software is completed to ensure any maneuvering through the mobile application comes with ease. It will be tested with outsides users. It should not be tested with only the team members of the project to avoid bias testing. Table 36 shows the three methods used to check for usability.

Table 36: Three methods used to check usability

Method	Description
Sketch	Draw a design of the mobile application and evaluate if it is appropriate.
Real Users	Use UCF students to use our mobile application and ask their opinion. May require strangers to avoid bias.
Team Critics	The Park Shark team’s evaluation on the mobile application usability. Quick and easy test.

8.2.2.2 Compatibility Testing

The compatibility testing is to check if the mobile application works properly with mobile devices, screen sizes, and OS versions. Unlike the web application the mobile application should ensure that is works for any screen size. Phone screens may be too small, so the design is taking this into consideration. The mobile application should be tested on iOS, windows, android, and black berry. Since this is a hybrid application it should be displayed in a very similar manner. The application will also be tested on multiple different screen sizes with restrictions to any device that has a screen size smaller than 3.61 in. Test will also check for operating systems that are no older than 2 years.

8.2.2.3 Integration Testing

Integration testing is checking to see if all components that should interact with each other are doing so correctly. Once components are integrated they are tested on the behavior and functionality. The major components of interest are integration of page transitions. There will be a continues testing on integration to ensure the mobile application is functional for all other testing purposes.

8.2.2.4 Memory Testing

The memory testing is checking to see how much memory is being used about the mobile application. Mobile devices do not have as much memory to spare as a desktop computer or laptop would, so the Park Shark team is testing to ensure the mobile application maintains an optimized memory usage as a component and while being used.

8.2.2.5 Performance Testing

The performance testing is checking the performance of the mobile application by taking a look at battery consumption, loading time, and stress. Although the mobile application is not expected to be opened for too long it should not be draining the users batter in a short amount of time or even at large intervals. This is being put to the test by allowing the mobile application to run for an hour and check its battery usage using the mobile devices settings option. The loading time is being tested when the mobile application is at normal usage and peak usage. It is tested based on how long it takes for the mobile application to respond to the user’s input. The consistence of response time is also being tested. We are testing how much the mobile application can take before reaching a breaking point. It will be tested on how it handles user input during and how it recovers from this state. The stress test is being done at small incremental of load.

8.2.2.6 Installation Testing

The installation testing is checking how the mobile application response to it being installed and uninstalled on a device. This is going to be tested by installing the mobile application onto multiple devices as well as uninstalling. Methods to check validation include checking the required memory space and after installation check the mobile devices setting to ensure the correct number is being displayed. When the mobile application is uninstalled the phone should recover the memory space that was being used. The mobile application will also be disrupted in the middle of

an installation to check the response. The system should recover and start back at its original state.

8.2.2.7 Security Testing

The security testing is used to detect potential vulnerabilities to the mobile application. The mobile like the web application will be providing the administrative login and this must be a secure component because it holds information that can be modified which would jeopardize the accuracy of the application. The validation of the username and password should be checked. There are restrictions to what passwords admins can use. Including a minimum length and requirement of special character. Testing of passwords that do not meet the length and character requirements should be rejected. The response to current user information and wrong user information is being tested. All wrong user information should be rejected with a warning displaying that either the username or password is incorrect.

9.0 Mounting and Installation

This section will detail the required steps to mount the system in the garage as well as steps required to setup finish installation and setup the final operation of the system.

One of the features the Park Shark system has to offer is the flexibility to mount the cameras and sensor modules where desired. The placement of the sensor system and computer vision system can be accommodated according to the specific layout of the parking garage.

The sensor system consists of individual wireless sensor modules. The number of sensor modules necessary for the parking garage depends on the layout of the customer's parking garage. The sensor device was created with the idea that one sensor module will be responsible for one parking space. The placement of the sensor module depends on the customer's preference and the parking garage's design. The sensor module's design allows it to be mounted on to two different places: the wall or the ceiling. These two choices are provided in other to provide the customer with an alternative when no ceiling or wall is present. For instance, the roof of the parking garage doesn't have a ceiling. The solution is to place the sensors on the walls. Some sections of the parking garage don't have any walls to mount the device, but a ceiling is available as a solution. The concept of the two mounting locations are shown below (Figure 44 and Figure 45).

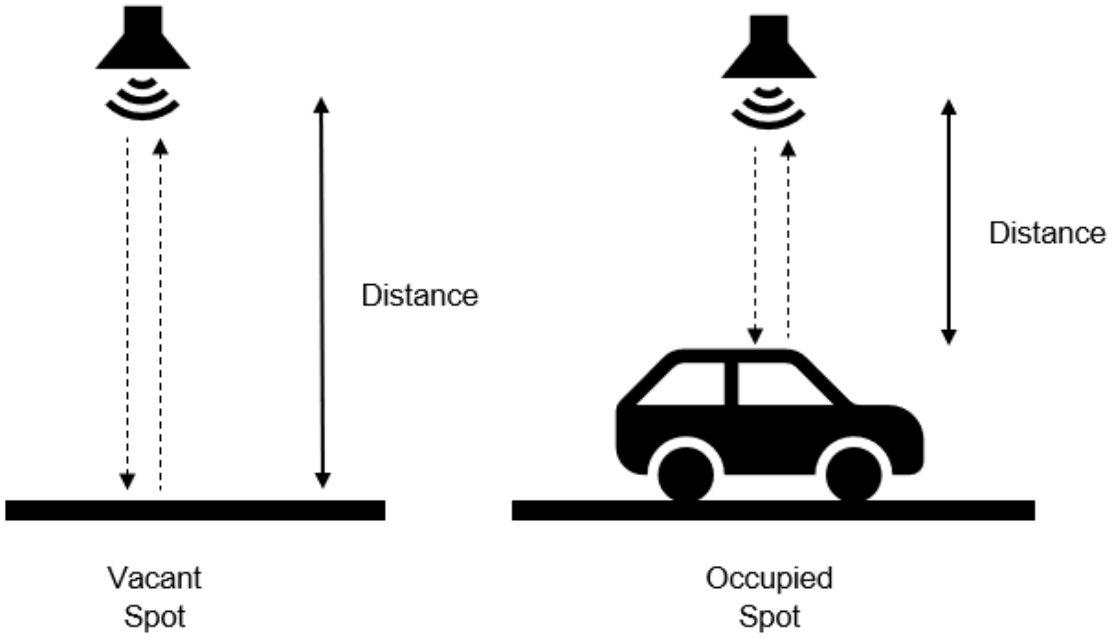


Figure 44: Demonstration of ceiling installation.

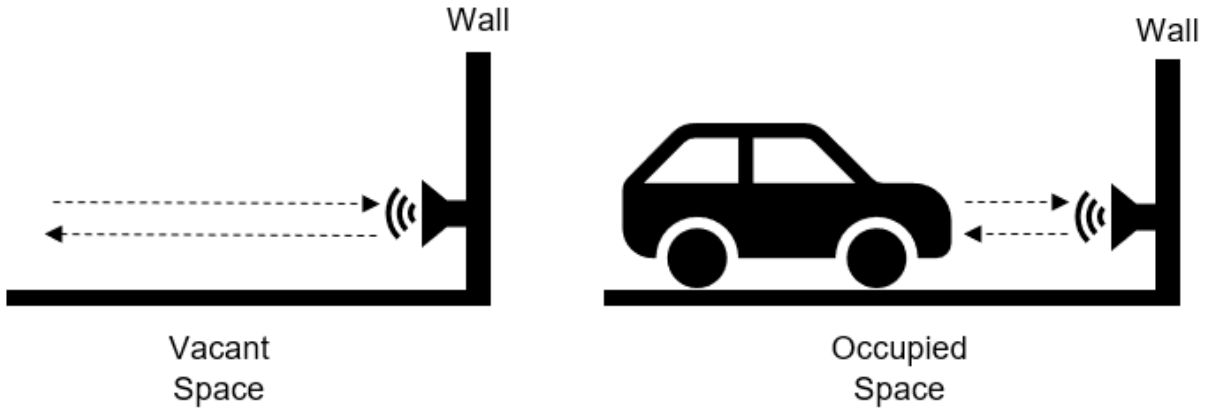


Figure 45: Demonstration of wall installation.

Some conditions must be taken into consideration when mounting the device onto either locations. If the sensor module is placed on the wall, the sensor should be placed high enough to decrease the risk of a vehicle's bumper hitting and damaging the module. If placed on the ceiling, the surface of the ceiling must be considered. Some parking garages have smooth ceilings while others might have ribbed ceilings. Also, when mounting the sensor module onto the ceiling, the owner must make sure it's properly secured. Most parking garages are made out of concrete so special tools and items will be necessary to properly mount the device onto the ceiling. One must make sure it's properly secured in order to prevent the situation of the sensor falling which could result in injury or damage to private property. Also, if the sensor is mounted to the ceiling, a reference of the distance between the sensor and the floor would be required. This way, when the sensor calculates a distance value smaller than the reference, it will signify that a vehicle has occupied the space.

The computer vision system consists a wireless camera module. Currently, the team believes each camera will be able to keep track of at least 3 parking spaces. The official number will remain as an estimate until the camera is purchased and tested. Ideally, the customer will place the camera at a location that grants it an optimal field of vision. The placement of the camera modules mostly depends on the layout of the garage and the owner's preference.

For the demonstration at the end of the semester, the idea is to section off a part of the UCF parking garage C (Figure 46) located on campus in order to display the system's functionality. Since the team does not have the funds to purchase equipment necessary to mount the sensor device onto the ceiling or wall, the camera and sensor modules will be mounted onto the wall using Velcro for demonstration purposes. At this time, the team is unsure as to how the cameras will be mounted onto the ceiling/wall of the parking garage. Other ideas on how to mount the equipment will be researched and will be thoroughly planned out by the start of Senior Design 2.

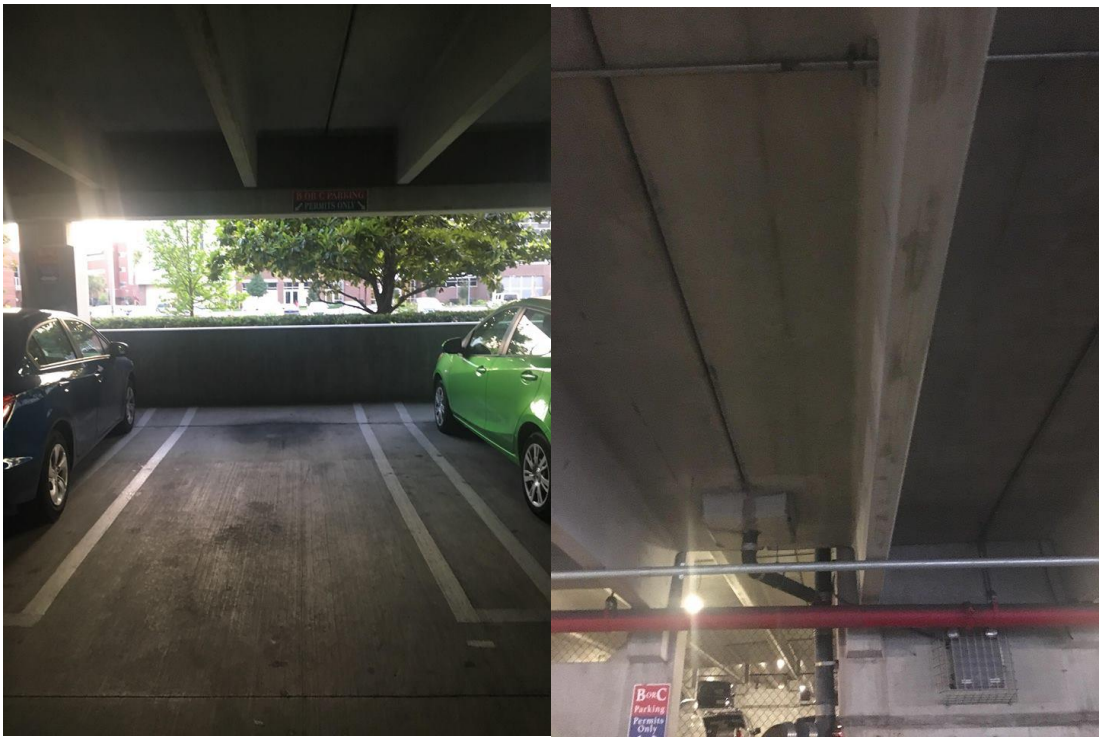


Figure 46: Wall and ceiling near Garage C parking spot.

Following installation, several tasks will have to be completed before operation of the sensors and web applications can be used. After the sensors have been mounted throughout the parking garage, the operator will ensure each has been powered and is currently running. The next step is to ensure connection between each individual Ultrasonic sensor is stable and that information is being sent to the master. The connection from the master ultrasonic sensor to the Computer Vision sensor (which is acting as the base communication for this project) needs to be ensured as well. Wireless connection via Wi-Fi will have to be setup as well. The operator will be required to SSH into the device and connect to an available network. If all data is being sent and the Computer Vision sensor is connected to the internet, then the Sensor system should be ready for operation. The use of the system will be completed and without need for any manual operation.

To access the website, users will visit the Park Shark link via a web browser and mobile users can download the Park Shark application via their respective mobile stores to access all provided information and functions. Once connected, the administrator will login to the admin account. The admin will be required to add the garage that the system was installed into the system and associate the sensor data being received with that garage. Once this is completed, the system will operate without need for any manual operation. Users will be able to login and access all pages without need for logging in to an account. All functions not related to the admin role, such as reporting average time and sending a message to the developer, can be accessed by users.

10.0 Project Management

Throughout the designing process as well as the implementation of our project, good project management will be necessary to ensure each member works well together. Without a good working environment setup and without proper communication between members of the team, the team would not be able to complete the project. This section will detail the tools used to ensure good team management as well as discuss the methods used to plan and manage the project.

The first tool used to help manage the project is Microsoft OneDrive. OneDrive is an online cloud storage service provided by Microsoft and implements functionality between itself and Microsoft's Office Suite. All member of the team preferred to use Microsoft Word to write their portions of the design, so one OneDrive was the obvious choice. The application allowed members to work on the design document together, uploading design portions and reviewing each section in real time. This allowed not only the ability for better collaboration but also allowed for the team to have a better understanding of the remaining portions of the design that needed to be completed.

The most important tool used within the team is Discord. Discord is a freeware VoIP application. The service allows users to create a channel for its users to talk and chat as well as share documents and figures in a fast and efficient manner. The service was used for all communication in the design phase, as well as used for all online meetings. Members in the channel also frequently connected to the chat outside of meetings to write their pages. This allowed members to ask questions or figure out design issues while each member was writing their respective portions of the project.

For Senior Design 1, the team held weekly 30-minute meetings where each member would discuss what each person had accomplished or not accomplished and detailed their goals and objectives for the following week. These meetings allowed members to have a better understanding of what was expected of them and allowed the team to gauge where the project design was at and adjust project schedules. These meetings were also used to air any concerns by the members for the project or other issues that may have arisen. This created an environment where members felt they were a member of the team and allowed them to make meaningful impact through discussion as well as taking on assignments.

The management of the team for Senior Design 2 will be very similar to the above style but will add scheduled build meetings in person on a regular basis. These meetings will be used to create prototypes, build any physical hardware required, and to carry out testing procedures. These meetings will be split into two categories, Software Build Meetings and Electrical Build Meetings. Depending on the members roll within the team, they will be required to attend their respective meeting, and attendance to the other meetings should be expected to ensure all members have a solid understanding of the project.

11.0 Administrative Content

This section will discuss the two main management components of the project. The two main management components being the financial plan and milestones of the project. This project has been broken up into two separate portions. The first one being research and prototyping. This portion focused on researching every component to the Park Shark system as well as determining which parts would work best for the systems performance while considering cost. After doing extensive research parts were bought and tested to check if results were satisfactory. The second portion of the project is for the next semester which is a list of tasks to create a working prototype, testing, and finalizing the project. A final presentation will also be required

11.1 Financial Plan

Since the project is being funded by the team members we wanted to ensure the project was affordable while still staying as functional as possible. Table 37 shows an estimate for each component in the system as well as the overall cost for the Park Shark system. Figure 47 is used to better display the cost of each component to the system.

Table 37: Estimations of cost.

Item	Unit Price	Quantity	Cost
Arducam 5 MP OV5647	14.99	2	29.98
Raspberry Pi 3 Model B+	35	2	70
16 GB Micro USB	8.45	1	8.45
AA Alkaline Battery	0.99	9	8.91
HC-SR04 + Ultrasonic Sensor	1.44	3	4.32
Nordic NRF52832	5.46	3	16.38
TPS62740 Step Down Converter	1.7	3	5.1
MAF94383 Antenna	1.39	3	4.17
Sensor Module Housing	10	3	30
Printed Circuit Board	1	10	10
Total Cost			$187.31

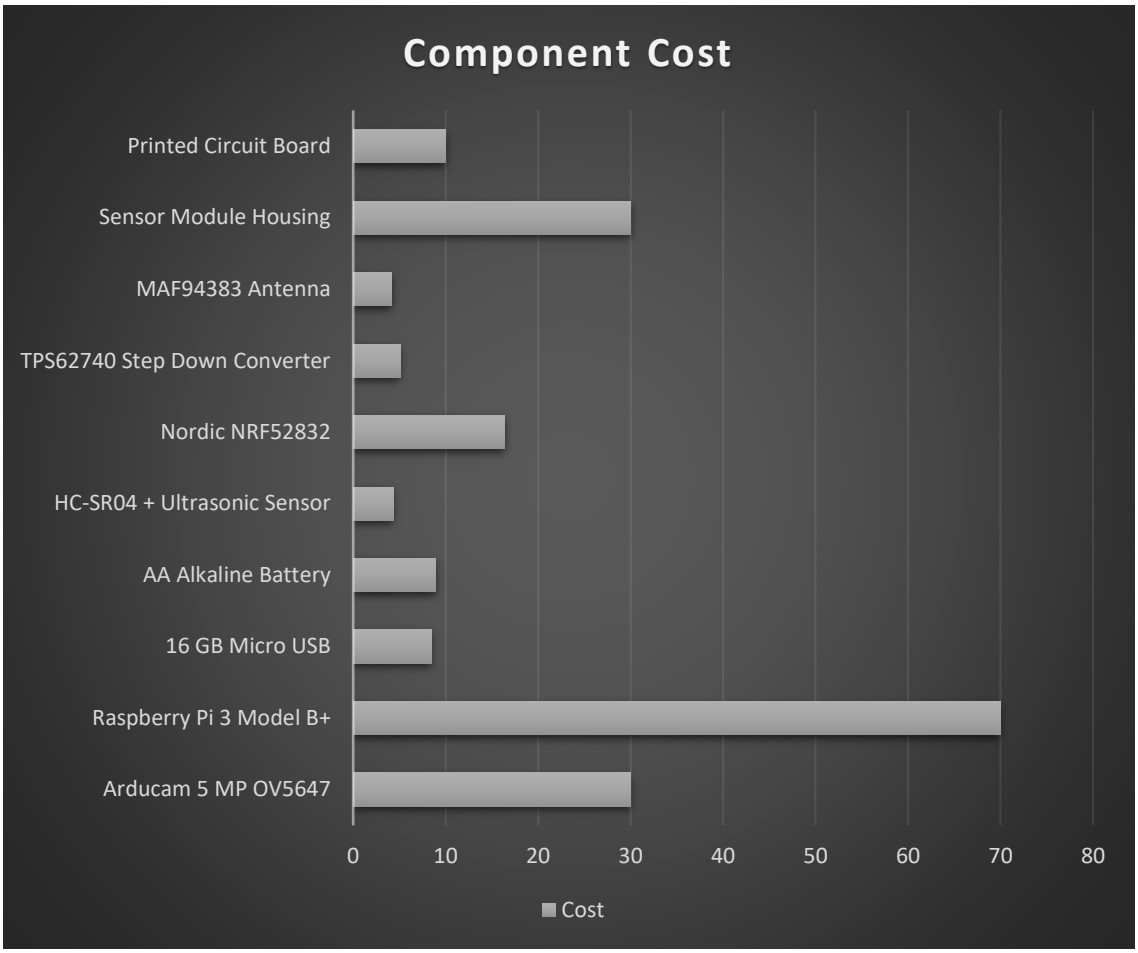
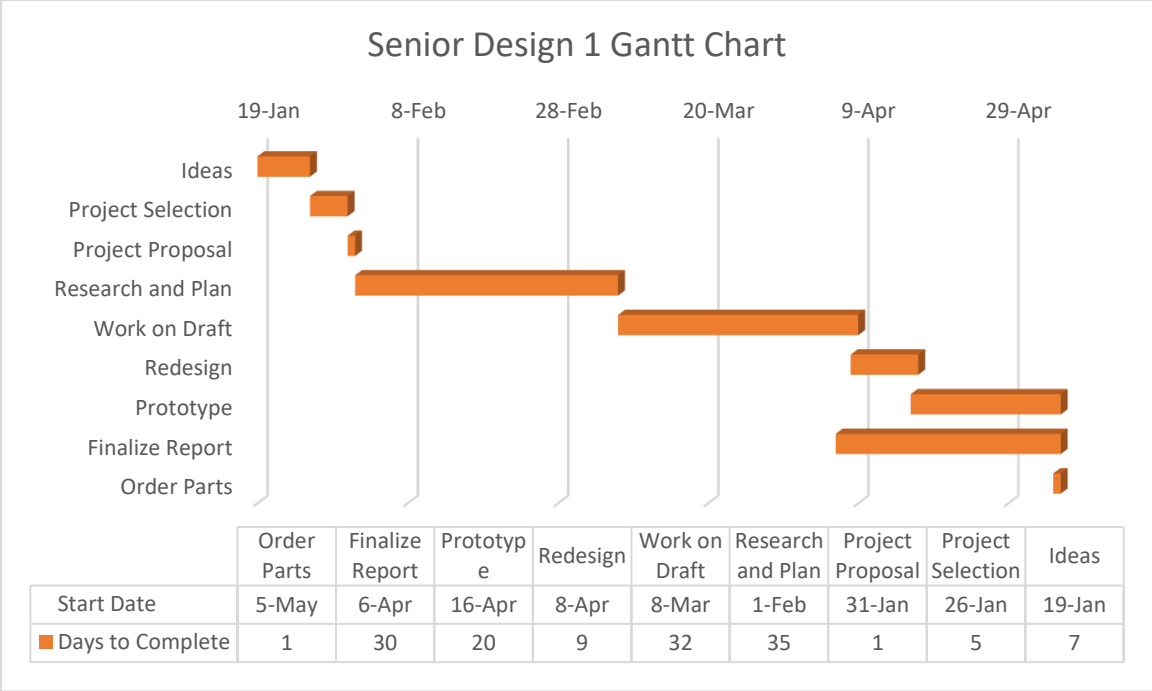


Figure 47: Visual representation of component cost.

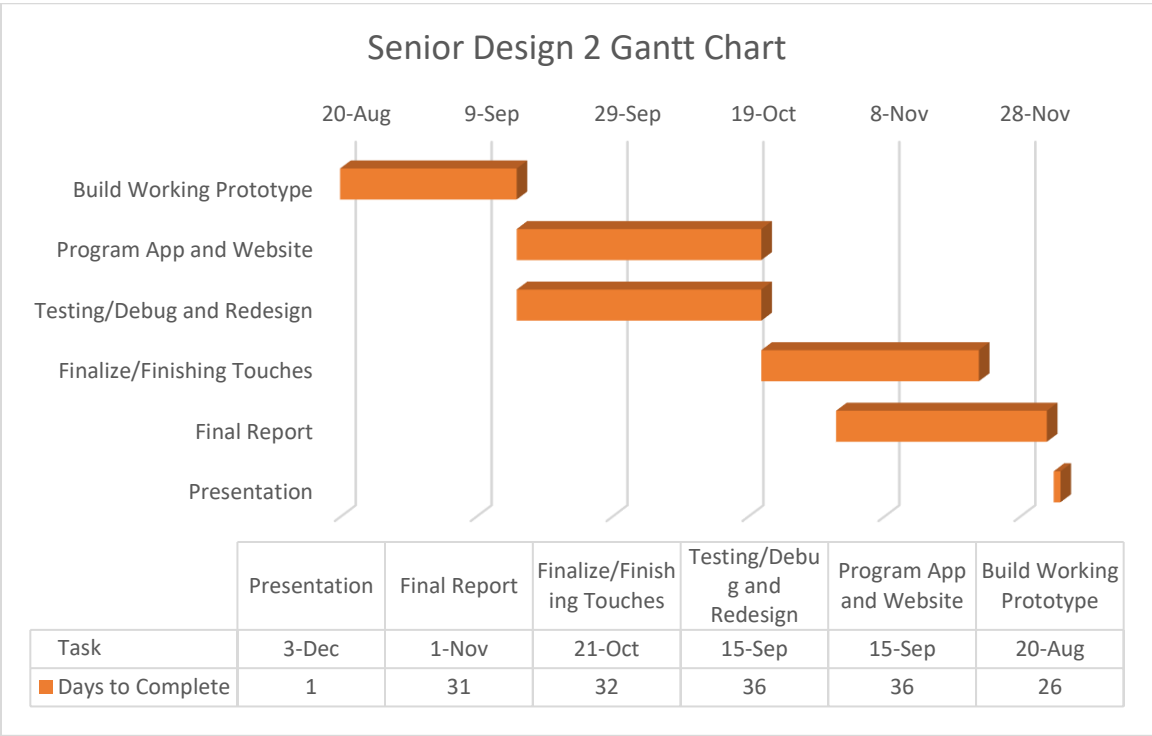
11.2 Milestones

The milestones have been broken into two separate portions: a Gantt chart showing the project timeline for Senior Design 1, and a second Gantt chart showing the project timeline for Senior Design 1. Senior Design 1 covers the research and design of the Park Shark project, as well as the construction of the design document. Senior Design 2 covers the prototyping and construction of the final Park Shark product, including programming, testing, and assembling of the products. The milestones have been chosen based on the assignment due dates required by Senior Design 1 and 2, as well as the availability schedules of each of the group members. The Gantt chart for Senior Design 1 is displayed in Figure 48, and the Gantt chart for Senior Design 2 is displayed in Figure 49. The milestones will have to be closely adhered to in order to ensure prompt completion of the product design as well as the product assembly and testing.



	Order Parts	Finalize Report	Prototype	Redesign	Work on Draft	Research and Plan	Project Proposal	Project Selection	Ideas
Start Date	5-May	6-Apr	16-Apr	8-Apr	8-Mar	1-Feb	31-Jan	26-Jan	19-Jan
Days to Complete	1	30	20	9	32	35	1	5	7

Figure 48: Gantt milestone chart for Senior Design 1.



	Presentation	Final Report	Finalize/Finishing Touches	Testing/Debug and Redesign	Program App and Website	Build Working Prototype
Task	3-Dec	1-Nov	21-Oct	15-Sep	15-Sep	20-Aug
Days to Complete	1	31	32	36	36	26

Figure 49: Gantt milestone chart for Senior Design 2.

12.0 Conclusion

This document represents the initial design for the Park Shark parking monitoring system. This section will serve as a total document summary and will review the important sections created, discuss some issues the team faced during design, and discuss some challenges we may face during the implementation process.

12.1 Document Summary

Sections 1 and 2 define the desired system as well as exact specifications and requirements of the system. The extensive research conducted in section 3 served as an area to explore different technologies that could be used to implement our system and they were extremely vital during the design phase. They also served as a great learning opportunity for the team to explore technologies that they may not have been familiar with. Section 4 highlighted the design constraints the system will have in terms of several criteria such as economic, environmental, and manufacturing. The design phase conducted in section 6 allowed the members to define and create engineering systems that will serve as great experience for each member as they near the end of their college career. The remaining sections served as exposure to the process of Systems Engineering, which team members had very minimal experience to, and will assist the members as they leave college and enter the work force where these processes are heavily utilized.

12.2 Issues Faced

During the design phase, several issues arose that, as a team, we wish to rectify in Senior Design 2. The first issue that arose is scheduled meeting attendance. The team initially intended to meet every Monday to have a 30-minute meeting to discuss design ideas, update the team with their progress, and discuss their plans for the coming week. While the team met frequently in the beginning of the semester, as final projects and exams neared the team did not meet or communicate at a satisfactory level. This will need to be rectified in Senior Design 2, as proper communication will be crucial in completing the design. The next issue was the tracking of work completed. Team members should be expected to complete a certain level of work within the team, and when individual members do not meet deadlines, it requires the entire team to work together to accomplish the needed tasks. For a project as large as this, individual work needs to be completed in a timely manner so that other members can accomplish their tasks without having to redirect their attention to meeting overdue deadlines. Task tracking tools such as Trello will be used in Senior Design 2 to ensure that each member of the team is completing their work on time.

12.3 Design Challenges

The implementation for this project will take place during Senior Design 2, and we will have three months to complete it. Given the design discussed in detail above, along with the associated requirements and project timeline, we feel very confident in our ability to accomplish this design. Project prototyping will occur in the first week of Fall semester, and any redesigns or major changes to the project will take place in that time. As the design created in the document is very rough, we expect

to be faced with numerous challenges. The main challenge will come in the creation of the PCB described in the Ultrasonic sensor design, and the implementation of the computer vision methods described in the Computer Vision sensor design. The members of this team have never created a PCB before, and while they have worked extensively with circuits and development boards, this will be a new experience for all members. The computer vision method used is one of the most advanced computer vision methods available today, and with its use comes complicated implementation and use within this project. The creation of our own dataset also creates variables of uncertainty, as the dataset could have large accuracy issues that will need to be tested against premade datasets. While we do feel confident, the prototyping phase will prove incredibly vital to ensure that the design described above can come to fruition.

12.4 Final Word

Senior Design 1 has been an unforgettable experience that has allowed each member to grow as an individual, gain experience working in a team, and learn new technologies that interest each member. As a team, we are very satisfied with the design, and are extremely excited to create a multi-sensor system and attempt to solve one of UCFs greatest issues: horrible parking.

13.0 Build Phase

During Senior Design II, the design of the project was put into practice and assembled into a working, physical system. This section describes the process of implementing the design, as well as all the difficulties and challenges encountered during the production of the project. Some aspects of design had to be altered somewhat in order to ensure a working system, however the specifications were all followed.

13.1 Backend Software

This section will summarize the implementation of the backend and frontend software portions of the project. These include the website and mobile application for the clients to access the system, as well as the software that delivers the data from the database to those frontends.

13.1.1 Server Backend

The original idea of creating the website with React and the mobile application with React-Native stayed the same, but the idea of using Redux to create a single component to talk to the backend changed. Instead all three components remained separate, but still worked as intended.

13.1.2 Website and Mobile Application

The website and mobile application initial design stayed mostly the same. The one key change was that the administration functionality was moved from the website to the mobile application. The new overall system design can be seen in the new architecture below in Figure 50.

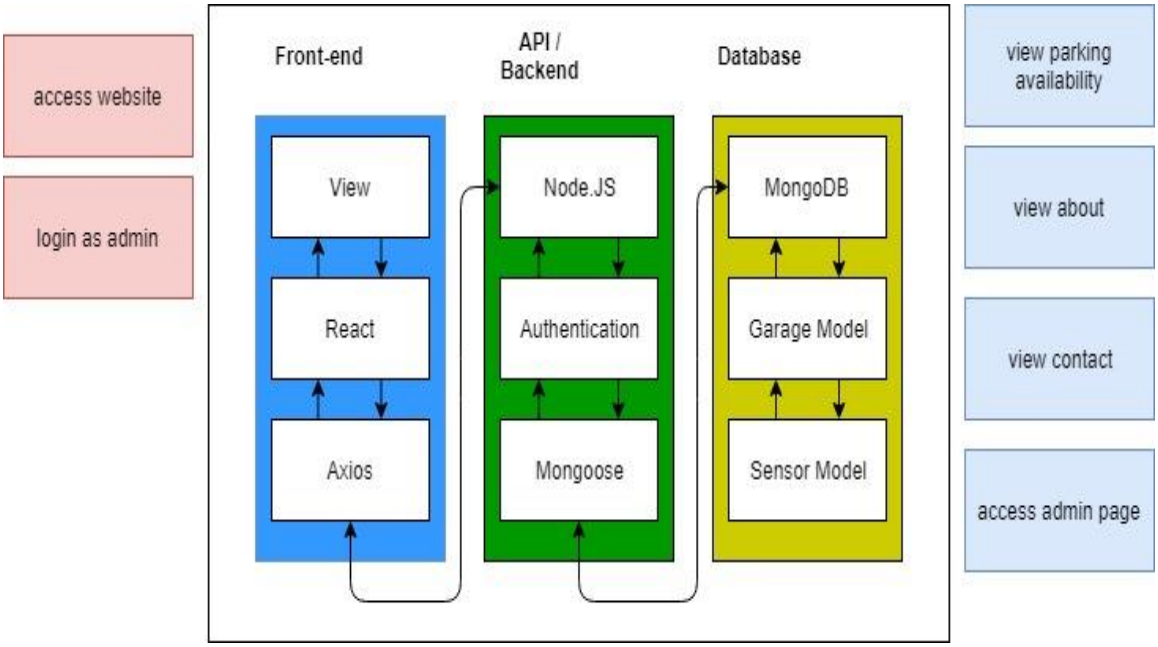


Figure 50: New API for frontend and backend.

13.2 Computer Vision Unit

This section is a summary of the issues that came about related to the computer vision unit and how they were solved.

13.2.1 Computer Vision Algorithms

In the initial research and design phase for the object detection algorithm, the group concluded that the use of the Tiny-YOLO model would be the best suited for the project. With a decreased runtime at the cost of accuracy, the assumption was that the loss of accuracy would not be critical to the detection of vehicles and the decreased runtime would allow us to update paring availability much faster. However, during the prototype phase and the subsequent testing phase, upon installing the camera in the garage the loss in accuracy of detection was much greater than anticipated. The model would detect several false positives as well as completely miss some detections. After noticing this effect, the decision to switch to the full model (specifically the newly released YOLOv3) was decided. While the algorithm did take a substantially longer time, the group still met their system requirements for the intervals they wanted to update parking availability. This increased runtime gave extremely high accuracy, which was more desirable overall.

Another issue also made itself apparent during the testing phase. Upon installing the camera, how could one define the spots you wanted to camera to look at, as well as the related problem of ensuring two camera units do not overlap and over count cars? The first attempt to solve this problem was the use of the SIFT algorithm. The idea was to take the images from the camera units, transfer them to one of the computer vision units, and utilize the SIFT algorithm to define features in the images and stitch the images together, resulting in a panoramic of the images. Then the YOLO algorithm would be ran on the resulting image, ensuring that we accomplished an accurate number of availabilities. Upon testing this method, it regrettably did not work as intended. Upon running the SIFT algorithm, not enough common features were detected between the two images, and therefore the resulting panoramic was inaccurate. The solution to this problem was to install the camera unit, view the resulting image, and crop the image to view the desired spots. While this solution was not particularly elegant, the result is 100% confidence in what spots a particular unit was looking at and therefore accomplishing accurate parking availability.

Upon the desire to introduce spot ID detection, the question of how to accomplish this with computer vision became apparent. The first attempt was to utilize the Hough Lines Transform to detect the white lines of the parking spots and utilize those lines to define the spots. However, several issues become immediately apparent after attempt to utilize this algorithm were made. The lines in the parking garage are often faded and hard to see from a camera, the lights in the garage would often wash out the lines, and from the POV of the camera, the lines could often not be observed from the computer vision unit. The solution to this problem was again to install the camera unit, view the resulting image, and manually define

the lines of the parking spots. While this solution again requires more manual work from the system installer, the result is 100% spot ID detection.

13.2.2 Raspberry Pi Bluetooth Library

One of the biggest challenges with the camera unit was getting the unit to read the data from the Bluetooth powered sensors. The team needed a way to read Bluetooth BLE data from the devices using Python. Considerable time was spent learning several different Python Bluetooth libraries (such as PyGATT, BluePy, GATT-Python, and PyBluez.) However, it was found that almost all these libraries had some missing functionality or were generally incomplete due to how relatively new the Bluetooth Low Energy communication standard is. For example, much time would be spent developing a data collector script using PyGATT only for it to be discovered later that the library does not support connecting to multiple devices at the same time, necessitating that the group move to using a different library in order to have a responsive enough system. The group was unable to get data notifications working using GATT-Python, meaning that the server must manually poll for new data instead of receiving it asynchronously, which would result in a higher power draw from the sensors to be constantly replying with the data, rather than only sending it explicitly when it has changed. Eventually the team settled on using the library BluePy, as it supported multiple connections. Though it required more lines of code than the other libraries, BluePy was the only one that supported all the Bluetooth Low Energy features that the project needed for optimal functionality.

13.3 Ultrasonic Sensing Unit

The wireless ultrasonic sensing unit underwent some design changes over the process of the build phase as the group had a better idea of what size they wanted the board to be, in order to have an effective and compact housing. Overall, the unit once constructed behaved as expected, with no significant issues other than struggles with refining the manufacturing and assembly method. The Nordic chip on the board was easily programmed using the Nordic development kit's debug passthrough pins just as planned. Although the group considered some changes that would somewhat improve the design, no second revision of the circuit board was necessary since all the wireless connection components functioned properly even on the very first constructed board.

13.3.1 Printed Circuit Board Redesign

The schematic of the circuit design, Figure 51, can be split into two main sections: the power supply and the MCU. The power supply section consists of the battery pack and the step-down voltage regulator. The battery pack is connected to the voltage regulator at pin P1 and supplies it with 4.5V. The voltage regulator converts the 4.5V to 3.3V necessary to power up the MCU and ultrasonic sensor. This voltage regulator is unique in the sense that it has two outputs: a constant 3.3V VDD output and a secondary switched 3.3V VCC output. This secondary output can be toggled by the nRF52 microcontroller via the VRM_CTRL pin to turn the sensor off when not in usage in order to conserve power. VRM_PG is a "Power Good" status pin indicating that the voltage regulator is correctly outputting 3.3V.

VRM_PG will turn Low when the output voltage starts dropping below 3.3V, indicating that the battery is getting low and should be replaced. VDD is used to power the Nordic chip, while VCC is used to power the ultrasonic sensor. The MCU section consists of the Nordic nRF52 chip and the ports for debugging and the sensor. Surrounding the microcontroller are a bunch of decoupling capacitors in order to filter out any background noise. There are also 2 crystals, a 32.768kHZ and a 32MHz, used to keep track of the timers. There is a 4-pin debug port connected at pin P4 that was used for programming the nRF52 microcontroller, as well as debug purposes. Port P2 on the microcontroller connects to the 4-pin header on the ultrasonic sensor. The schematic was created using the EagleCAD software.

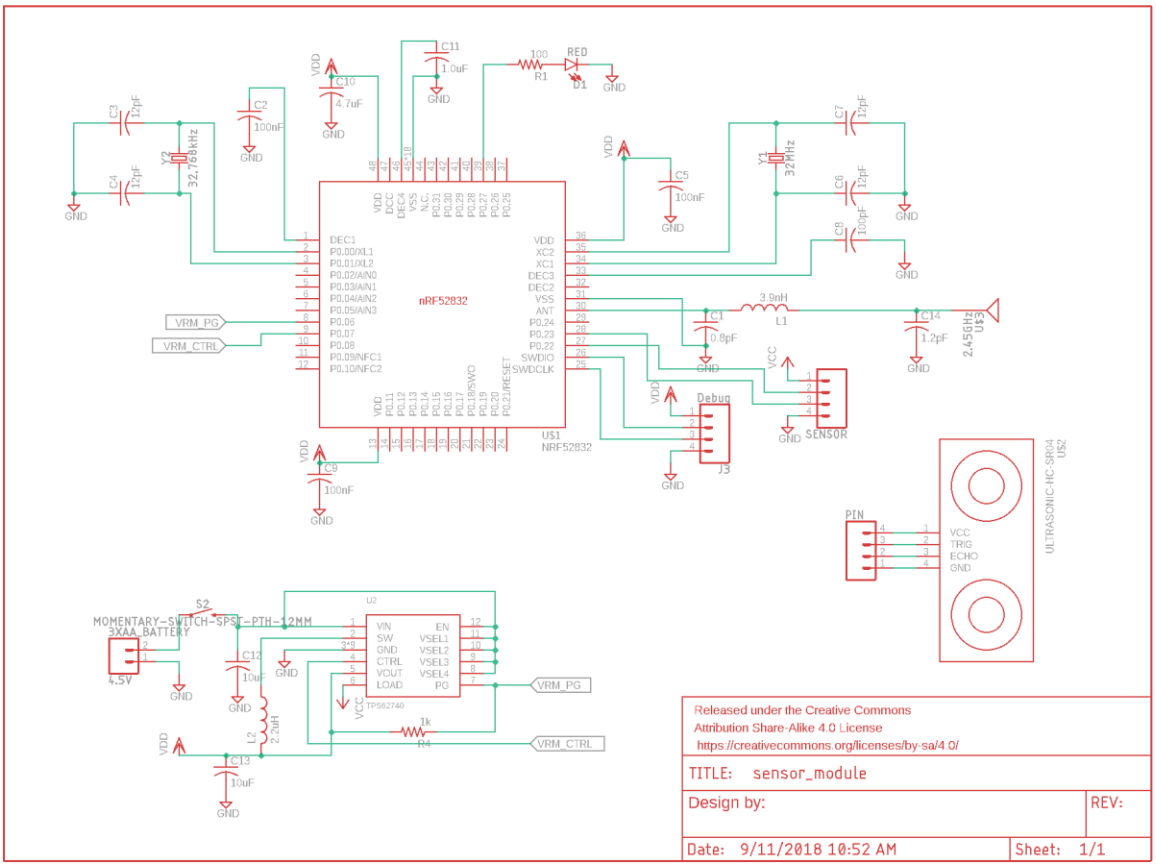


Figure 51: Schematic for the sensor module.

Once the schematic was made, the EagleCAD software was used to convert it into a PCB board layout as shown in Figure 52. All the components were manually connected following PCB layout design rules and suggestions. The board has a ground plane on both sides, which was recommended by the nRF52 reference circuits. The reference sheet also suggested to place the components as close as possible to the microcontroller and to place each one at a certain location. Thus, SMD components were used in order to make the PCB smaller in size and to bring the components closer to the MCU as suggested.

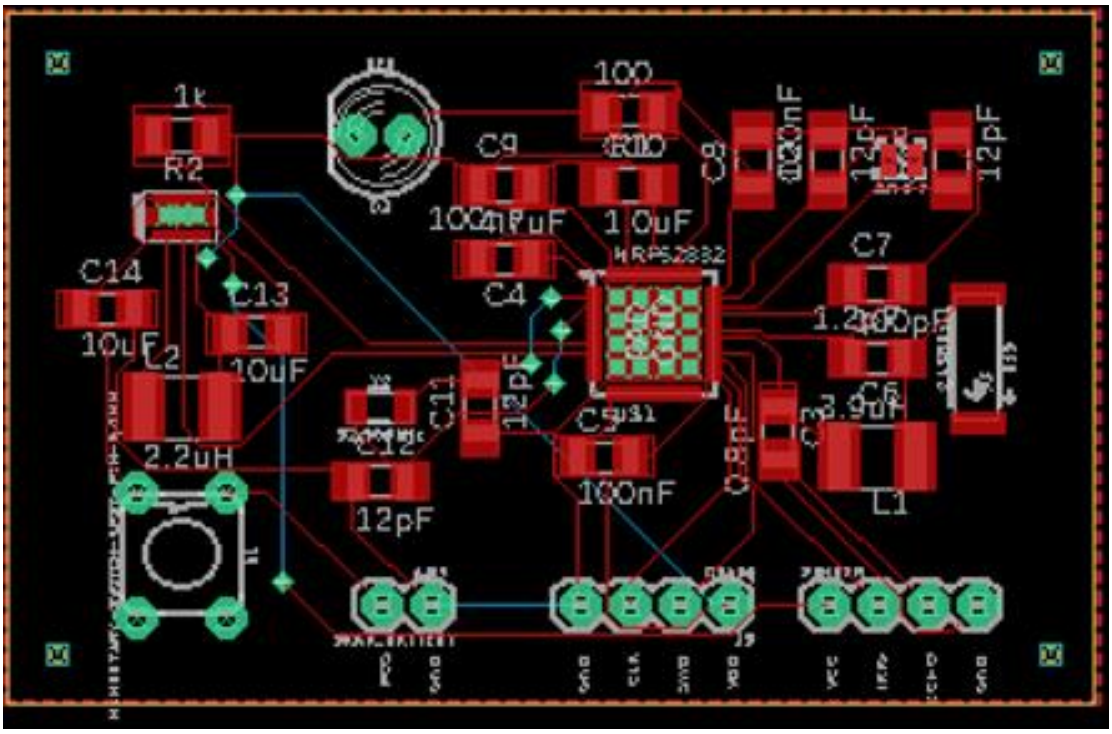


Figure 52: PCB layout for the sensor module.

13.3.2 Printed Circuit Board Manufacturing

After purchasing and receiving all the components for the PCB board, it was time to solder them onto the PCB. Since the circuit board is mostly comprised of SMD components, a stencil was purchased to make the soldering process a bit easier. A stencil of the PCB layout was purchased from OSH Stencils for a total of $18.67. The SMD components were soldered onto the PCB using solder paste, specifically SMD297SNL50T3 SAC305 No-Clean T3 solder paste from CHIPQUIK. Before placing the stencil, the PCB and stencil were cleaned using isopropyl alcohol and dusted off to remove any debris. The PCB was then taped down on the table to prevent it from moving. The stencil was also taped down once it was carefully positioned on the PCB. A blob of solder paste was then placed on the stencil and spread over the PCB using a plastic card. The stencil was carefully removed revealing an evenly coated PCB. Before placing the components, the pins were examined up close to make sure no pins were bridged, especially for the microcontroller and voltage regulator. Each individual component was placed at their corresponding locations using a pair of tweezers, while carefully following the PCB layout making sure not to place the wrong component in the wrong location. After all the components were mounted onto the board, the PCB was placed onto a hotplate set to 270 degrees Celsius for about 5 minutes until the solder paste completely melted and solidified. After inspecting the pins for each component, the one problem we would encounter was the bridging between some of the pins for the microcontroller. We easily solved this problem by swiping the soldering iron tip along the pins with a bit of flux which successfully unbridged the pins. After the SMD components were soldered, the through hole components were next. Female

head connectors were used for the power supply, debug port, and sensor port. The connectors and red LED were soldered onto the board using a soldering iron and lead-free rosin core solder. In addition, a small wire was soldered where a switch was originally going to be placed. A separate switch was no longer necessary since the battery holder already had a built-in switch. The resulting circuit board is shown in Figure 53.

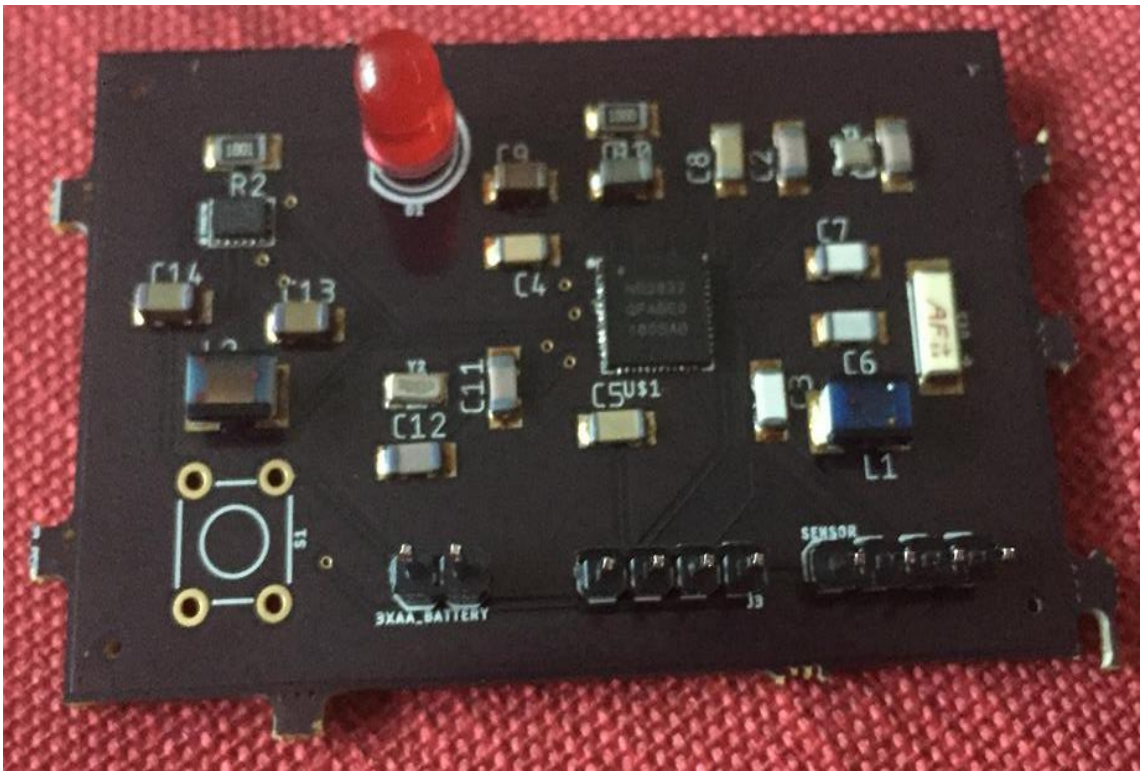


Figure 53: Final PCB for sensor module.

While testing the PCB, there was small problem we encountered. Due to accidentally applying 12V to the 4.5V input terminals while testing the low battery notification feature of the device, one of the voltage regulators on one of the boards had to be replaced. The voltage regulator internally shorted itself, possibly intentionally in order to protect the rest of the circuit. In the future, an overvoltage protection built into the board could result in a much easier repair in the case of accidental overvoltage conditions. However, in practice, this condition should never arise outside of purposely using a variable power supply instead of using the specified 3x AA batteries to power the device.

Table 38 on the next page lists the cost of each component comprising the final PCB layout for the sensor module. All the components were purchased from Amazon, Mouser, Digikey, and Sparkfun. It took about a week for all the components to arrive. The inductors were size 1812 while the capacitors and resistors were size 1206. The PCB was manufactured by OSH Park at a price of 3 boards for $15. It took about two weeks for the PCB to be made and shipped.

Table 38: List of components for the sensor module.

Item	Component Cost	Quantity	Total Cost
Nordic NRF52832	$5.33	1	$3.51
TPS62740 Voltage Regulator	$1.98	1	$1.98
HC-R04+ Ultrasonic Sensor	$1.44	1	$1.44
2.4GHz Chip Antenna	$2.95	1	$2.95
Red LED	$0.06	1	$0.06
Pin Head Connector	$0.02	3	$0.08
32 MHz Crystal	$0.75	1	$0.75
32.768 kHz Crystal	$0.50	1	$0.50
SMD Resistors	$0.02	2	$0.04
SMD Capacitors	$0.10	14	$1.50
SMD Inductors	$0.74	2	$1.48
3xAA Battery Holder	$2.64	1	$2.64
PCB Board	$5.10	1	$5.10
Total Cost		30	**$21.73**

13.3.3 Sensor Housing

Initially, the system housing for the sensor module was going to be 3D printed at the TI lab at no cost. Unfortunately, due to time constraint and lack of material, the system housing was not 3D printed. Luckily, we came up with an alternate solution. Since we had a CAD of the housing unit and knew the dimensions for the unit, we decided to use the laser cutter available at the TI lab to make the box. The housing unit was made free of cost because it was made from the scraps of wood available at the lab. Overall, the housing unit met the size and weight requirements that were initially set for the system housing. The final housing unit was shown in Figure 54.

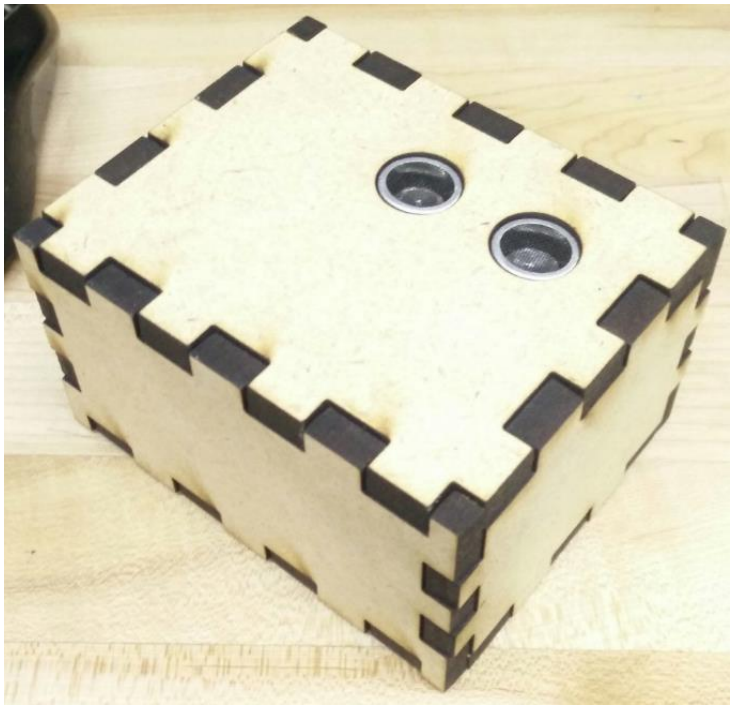


Figure 54: System housing for the sensor module.

13.3.4 Power Draw

The current draw for the ultrasonic sensor unit was much higher than predicted. The initial design theorized the average current draw would be somewhere around 0.05 milliamps, however in practice the device draws an optimal power draw of 0.2 to 0.3 milliamps on average. The reason for the discrepancy is that the Nordic online power draw estimator only shows the power draw of the Bluetooth radio transmissions themselves, and not the current required for the ARM CPU itself to operate. Additionally, other components such as the clock oscillator crystals have their own power draw.

Nonetheless, due to intentionally overshooting the power requirements necessary for the device to last one year on battery power, the resulting 0.3 mA power draw is still sufficient enough for the device to theoretically last 368 days. This cannot be verified due to the several months length of time of the project.

As predicted, the power draw average power draw is primarily lowered by changing the interval at which the ultrasonic sensor is powered and sends a sound ping. When the sensor is unpowered and not pinging, the wireless sensor device was measured to draw just .1 milliamps of current. Whenever the sensor activates and pings, the current spikes to about 3-4 milliamps. Due to the very small duty cycle of the sensor pings due to the interval system described in the design, these spikes in current draw do not significantly raise the average power draw.

13.3.5 Firmware

To write the C firmware for the wireless ultrasonic sensing unit, the Nordic nRF52 SDK was used. This provides the functions used to access the Bluetooth hardware

of the Nordic nRF52 chip through simple C function calls. First, a program was created to report only the distance measured by the ultrasonic sensor. Then, this was adapted into reporting whether or not a spot is occupied. Afterward, additional functionality such as low battery warnings were implemented. Finally, the dynamic sleep interval system described in the design portion was implemented and the Bluetooth connection intervals tweaked in order to ensure the power draw was low enough to suit the targeted battery life. The firmware was flashed to the circuit board's microcontroller using the Nordic nRF52 dev kit's debug passthrough pins, in order to avoid having to purchase the much costlier SEGGER J-Link to do the same task.

13.4 Final Budget

Throughout the course of the semester, various components were purchased as we worked on the monitoring system. Table 39 shows the cost for each component in the Park Shark system as well as the cost of the components used during the development of the overall system. The overall development cost came out to be more than we expected.

Table 39: System housing for the sensor module.

Item	Component Cost	Quantity	Total Cost
Arducam 5 MPOV5647	$14.99	1	$14.99
Pixy Cam	$69.00	1	$69.00
Raspberry Pi 3 Model B+	$35.00	1	$35.00
DROK Voltage Converter	$12.16	1	$12.16
12V Battery Pack	$4.00	2	$8.00
16 GB Micro USB	$8.45	1	$8.45
48 x AA Alkaline Battery	$18.98	1	$18.98
Nordic Dev Board	$39.00	1	$39.00
PCB	$21.73	3	$65.19
PCB Stencil	$18.67	1	$18.67
Solder Paste	$16.95	1	$16.95
Rosin Core Solder	$8.27	1	$8.27
Module Housing	FREE	FREE	FREE
Total Cost			**$314.57**

Table 40 and Table 41 list the cost of each component that makes up each individual unit. In the end, the cost to build a sensor unit is much cheaper than building a computer vision unit.

Table 40: Cost of sensor unit.

Item	Unit Cost
3 x AA Alkaline Battery	$1.19
PCB	$21.73
Module Housing	FREE
Total Cost	**$22.92**

Table 41: Cost of computer vision unit.

Item	Unit Cost
Arducam 5 MP OV5647	$14.99
Raspberry Pi 3 Model B+	$35.00
DROK Voltage Converter	$12.16
12V Battery Pack	$4.00
16 GB Micro USB	$8.45
8x AA Alkaline Battery	$4.75
Module Housing	FREE
Total Cost	**$79.35**

13.5 Final Conclusion

At the end of this project, and upon reflecting on the final product, a very clear flaw was present with utilizing computer vision to monitor parking. The amount of design work that went in to the computer vision algorithms, and some of the unfortunate solutions that were used to solve issues with defining the view of the camera and spot ID detection, the question of whether the computer vision unit was a good solution for this problem arose within the team. At the beginning of this project, the group had a hypothesis. Can one utilize a camera to look over more than one spot and be more efficient then hardware dedicated to each spot? At the end of this project, and from the opinion of the team, the answer they came to is: not yet. The computer vision unit utilizes advancements in computer vision to accomplish a task that was easily accomplished with relatively simple ultrasonic hardware. The computer vision unit is also more expensive then the production of an ultrasonic unit. For the number of spots monitored by the computer vision unit,

one could produce the same amount of the ultrasonic sensors for each spot and accomplish 100% parking availability accuracy. The impression of the developer of the computer vision unit at the end of this project was that they were attempting to utilize complicated software to replace simple hardware. If the group was to attempt to market this system to universities or companies, serious consideration of utilizing the ultrasonic sensor as a sole means of detection should be made.

After the project was completed, the team was incredibly satisfied. While the group may not have accomplished the most efficient system, the amount of knowledge and experienced gained through this project has been invaluable. The team also operated extremely well, with no major issues in accomplishing work, or personal issues within the team. The group worked together professionally to accomplish its goals and resolve issues. All the members of the Park Shark team have no doubt that the experience gained from this project will be useful in their future careers.

A. Appendix A: References

[1]cmucam.org/projects/cmucam5/wiki/Latest_release

[2]https://github.com/ArduCAM/Arduino (#2)
[3]https://github.com/ArduCAM/Arduino.git (#3)

[4]https://github.com/dasaaki/arduvision (#4)

[5]http://www.cmucam.org/projects/cmucam5/wiki/Hooking_up_Pixy_to_a_Raspberry_Pi

[6]Restuta GitHub / Created Feb 8, 2018 Angular and React sizes. Retrieved April 20, 2018, from https://gist.github.com/Restuta

https://sensing.honeywell.com/hallbook.pdf

https://www.electronics-tutorials.ws/electromagnetism/hall-effect.html

https://howtomechatronics.com/how-it-works/electrical-engineering/hall-effect-hall-effect-sensors-work/

https://www.youtube.com/watch?v=Scpi91e1JKc

https://www.allaboutcircuits.com/technical-articles/understanding-and-applying-the-hall-effect/

https://www.sans.edu/cyber-research/security-laboratory/article/bluetooth

http://www.rfwireless-world.com/Tutorials/Bluetooth-frequency-allocations.html

https://electronics.howstuffworks.com/bluetooth2.htm

http://www.mt-system.ru/sites/default/files/docs/documents/bluetooth_le_comparison.pdf

https://www.thedroidsonroids.com/blog/bluetooth-classic-vs-bluetooth-low-energy-on-android-hints-implementation-steps

http://www.rfwireless-world.com/Terminology/Bluetooth-vs-BLE.html

https://www.bluetooth.com/bluetooth-technology/radio-versions

https://www.rtings.com/headphones/learn/what-is-bluetooth

http://www.lotusgemology.com/index.php/library/articles/294-ftir-in-gem-testing-ftir-intrigue-lotus-gemology

https://www.elprocus.com/communication-using-infrared-technology/

https://learn.sparkfun.com/tutorials/ir-communication

http://www.rfwireless-world.com/Tutorials/Infrared-Communication-IrDA-tutorial.html

https://paginas.fe.up.pt/~ee05005/tese/arquivos/wireless_ir_com.pdf

http://education.rec.ri.cmu.edu/content/electronics/boe/ultrasonic_sensor/1.html

http://www.symmetron.ru/suppliers/murata/files/pdf/murata/ultrasonic-sensors.pdf

https://www.elprocus.com/different-types-of-arduino-boards/

https://learn.sparkfun.com/tutorials/what-is-an-arduino

https://www.elexp.com/PDFs/01ARDUNO.pdf

https://computer.howstuffworks.com/wireless-network1.htm

http://www.pearsonitcertification.com/articles/article.aspx?p=1329709&seqNum=4

http://literature.rockwellautomation.com/idc/groups/literature/documents/td/prox-td001_-en-p.pdf

https://www.cui.com/catalog/resource/power-supply-safety-standards-agencies-and-marks.pdf

Woodford, Chris. (2007/2017) Wireless Internet. Retrieved from http://www.explainthatstuff.com/wirelessinternet.html.

Brain, Marshall. “How Lithium-Ion Batteries Work.” HowStuffWorks, HowStuffWorks, 14 Nov. 2006, electronics.howstuffworks.com/everyday-tech/lithium-ion-battery.htm.

Menke, Henry. “Basic Operating Principle of an Inductive Proximity Sensor.” AUTOMATION INSIGHTS, 1 Feb. 2017, automation-insights.blog/2014/03/05/basic-operating-principle-of-an-inductive-proximity-sensor/.

AZoSensors, Written by. “Hall Effect Sensors.” AZoSensors.com, 27 July 2017, www.azosensors.com/article.aspx?ArticleID=16.

“BU-106: Advantages of Primary Batteries.” Lithium-Based Batteries Information – Battery University, batteryuniversity.com/learn/article/primary_batteries.

Projects, Microcontrollers. “Pixy Cam to with Arduino.” YouTube, YouTube, 29 July 2015, www.youtube.com/watch?v=DV4YK_Kk5IY&t=124s.

Jackson, Lee. “Arducam 2MP Mini Camera for Arduino Tutorial 2018.” YouTube, YouTube, 18 Jan. 2018, www.youtube.com/watch?v=hybQpjwJ4aA&t=305s.

Kirbis, David Sanz. “Arduvision (II): OV7670 FIFO Module and Arduino Mega.” TheRandomLab, therandomlab.blogspot.com/2016/06/arduvision-ii-ov7670-fifo-module-and.html.

“Hybrid vs Native Mobile Apps - The Answer Is Clear.” YML | Products and Experiences with Lasting Impact, 19 Oct. 2018, ymedialabs.com/hybrid-vs-native-mobile-apps-the-answer-is-clear.

Media, Traversy. “Mobile Apps - Web vs. Native vs. Hybrid.” YouTube, YouTube, 4 Aug. 2017, www.youtube.com/watch?v=ZikVtdopsfY.

“Pixy – PixyCam.” PixyCam, charmedlabs.com/default/pixy-cmucam5/.

“CMUcam5 Pixy.” Overview - CMUcam5 Pixy - CMUcam: Open Source Programmable Embedded Color Vision Sensors, www.cmucam.org/projects/cmucam5.

“OpenCV Integrate in MCU Edit.” Fingerprint Matching in Mobile Devices (Android Platform) - OpenCV Q&A Forum, answers.opencv.org/question/119854/opencv-integrate-in-mcu/.

“Infrared Night Vision Surveillance Camera 2-Pcs Infrared Lights for Raspberry Pi.” SainSmart.com, www.sainsmart.com/products/infrared-night-vision-surveillance-camera-2-pcs-infrared-lights-for-raspberry-pi.

“ArduCAM USB Camera Shield Released.” Arduino Based Camera, 17 Mar. 2018, www.arducam.com/arducam-usb-camera-shield-released/.

“2MP: OV2640.” Arduino Based Camera, 18 Mar. 2018, www.arducam.com/camera-modules/2mp-ov2640/.

“5MP: OV5642.” Arduino Based Camera, 18 Mar. 2018, www.arducam.com/camera-modules/5mp-ov5642/.

“Home.” Bitlunis Lab, bitluni.net/ov7670-with-fifo/.

B. Appendix B: Permissions

RH

Wed 3/28/2018 12:41 PM

Ryan Hildebrand <ryanh@signal-tech.com>

Re: Permission for use - University of Central Florida

To Keegan Conway

You sure can, Thank you very much for asking before!

Good luck!

Ryan Hildebrand
Inside Sales | ryanh@signal-tech.com

Signal-Tech
4985 Pittsburgh Ave. | Erie, PA 16509
Phone: 814-835-3000 | Toll Free: 877-547-9900 | Fax: 814-835-2300

Wholesale Manufacturer of LED Signs & Signals
www.signal-tech.com | Banking | Parking | Highway | Rail | Institutional & Industrial
Facebook | Twitter | LinkedIn | Google+

On 3/28/2018 12:38 PM, Keegan Conway wrote:

> Hello,
>
> My name is Keegan Conway. I am a senior at the University of Central Florida studying computer engineering. I am currently working on a parking garage monitoring system for my Senior Capstone project. A required section for our project documentation is a comparison section that compares existing, commercial products to our design. I would like to ask for permission to use some of the technical information and images found in your RedStorm 2.1 Parking Guidance System technical manual for use in comparison of technologies. The use of this information will be used for academic purposes only and will not modify or alter any aspects of your system.
>
> Thank you for time,
> Keegan Conway
> conwaykc@knights.ucf.edu

SR

Thu 4/12/2018 6:11 AM

Samuel Richie <richie@ucf.edu>

Re: Senior Design 1 - Group D - Power Supply Question and Image Permission

To Keegan Conway

The answer is yes to both questions.

> On Apr 11, 2018, at 10:08 PM, Keegan Conway <conwaykc@Knights.ucf.edu> wrote:
>
> Hello Dr. Richie,
>
> My name is Keegan Conway, and I am member of Group D (parking garage monitoring group). I would like to know if it would be possible to set a constraint on the power supply for our computer vision sensor system. As it currently stands we will be utilizing a Raspberry Pi and a camera to use OpenCV to detect cars. We will also be using this system to receive data from our hall effect sensors via Bluetooth, and transmitting that data to the internet. Our team wanted to know if it would be possible to use a battery pack to power this system, setting this as one of our system constraints. In a real scenario we would require the customer to supply power to the system via preinstalled power lines.
>
> I would also like to ask for permission to use the attached image in our paper, which was used in an example of computer vision detection methods.
>
> Thank you,
> Keegan Conway
> ke264290
> conwaykc@knights.ucf.edu
>
> <img_richie.jpg>

Re: Permission for use - University of Central Florida

SparkFun Customer Service <cservice@sparkfun.com>

Fri 4/20/2018 3:23 PM

To: Beatriz Jimenez <beaj1195@Knights.ucf.edu>;

Type your response ABOVE THIS LINE to reply

Beatriz Jimenez
Subject: Permission for use - University of Central Florida

APR 20, 2018 | 01:19PM MDT
Annabel B. replied:

Hey Beatriz!

We are totally open source, so you're welcome to use our images as long as you give us credit for them!

If you need anything else or have any other questions, let us know.

Have a great weekend!

Annabel Bonner
Customer Service
SparkFun Electronics
303.284.0979

APR 20, 2018 | 01:04PM MDT
Original message
Beatriz wrote:

Hello,

My name is Beatriz Jimenez and I am a senior at the University of Central Florida studying electrical engineering. I am currently working on a parking garage monitoring system for my senior design project. I would like to ask for your permission to use some of the pictures of your sensor products and microcontroller devices and utilize information about their features. This information will be used for academic purposes only and will not modify or alter any aspects of your products.

Thank you for your time,
Beatriz Jimenez
beaj1195@knights.ucf.edu

Re: Permission for use - University of Central Florida

BatteryU <BatteryU@cadex.com>

Wed 4/25/2018 1:33 PM

To: Beatriz Jimenez <beaj1195@Knights.ucf.edu>;

Hi Beatriz,

Yes, you may use the material as requested. Please cite source where appropriate.

Regards,

John Bradshaw - Marketing Communications Manager
Cadex Electronics Inc. | www.cadex.com
Vancouver | Minneapolis | Frankfurt
Tel: +1 604 231-7777 x319 | Toll Free: 1-800 565-5228

Follow us on Twitter: twitter.com/cadexelectronic
Join us on Facebook: facebook.com/cadexelectronics
Add us on Google+: plus.google.com/+Cadex

>>> Beatriz Jimenez <beaj1195@Knights.ucf.edu> 4/25/2018 9:13 AM >>>
Hello,

My name is Beatriz Jimenez and I am a senior at the University of Central Florida studying electrical engineering. I am currently working on a parking garage monitoring system for my senior design project. I would like to ask for your permission to utilize the tables, pictures, and information about batteries provided on your website. This information will be used for academic purposes only and will not modify or alter any aspects of your products.

Thank you for your time,
Beatriz Jimenez
beaj1195@knights.ucf.edu