CS 481

Park-IT-CdA Design Report

Nikolai Tiong

Tyrel Parker

Zane Goodrich

Joel Berain

Table of Contents

[Executive Summary 3](#_Toc40465099)

[Background 4](#_Toc40465100)

[Problem Definition 4](#_Toc40465101)

[Project Plan 6](#_Toc40465102)

[Team Members 6](#_Toc40465103)

[Concepts Considered 7](#_Toc40465104)

[Wireless Transmission 7](#_Toc40465105)

[Microcontroller 7](#_Toc40465106)

[Battery and Power 8](#_Toc40465107)

[Enclosure 10](#_Toc40465108)

[Simulation 10](#_Toc40465109)

[Encryption 10](#_Toc40465110)

[Concept Selection 12](#_Toc40465111)

[Wireless Transmission 12](#_Toc40465112)

[Microcontroller 12](#_Toc40465113)

[Battery 12](#_Toc40465114)

[Enclosure 12](#_Toc40465115)

[Sensors 12](#_Toc40465116)

[Simulation 15](#_Toc40465117)

[Encryption 15](#_Toc40465118)

[System Architecture 16](#_Toc40465119)

[Design Evaluation 18](#_Toc40465120)

[Vehicle Detection and Indication 18](#_Toc40465121)

[LED Indication 18](#_Toc40465122)

[Hardware Enclosure 18](#_Toc40465123)

[Mesh Network 18](#_Toc40465124)

[Data Collection 19](#_Toc40465125)

[Data Analysis 19](#_Toc40465126)

[Testing Results 19](#_Toc40465127)

[Future Work 20](#_Toc40465128)

[Appendices 21](#_Toc40465129)

[Battery Power Calculations 22](#_Toc40465130)

[Original Schedule 24](#_Toc40465131)

[Final Schedule 25](#_Toc40465132)

[Shared Folder Structure 26](#_Toc40465133)

# Executive Summary

Too much time and effort are spent by those looking for a place to park in one or more garages in downtown Coeur d’ Alene. Commuters would benefit greatly from a system that indicates whether there are spaces available for parking within a parking garage. The Garage Sensor System (GSS) will allow people to find open parking spaces, and hopefully provide this information before they enter the garage. The Garage Sensor Units (GSU) will be placed above parking stalls to indicate the status via LEDs; a green light means there is an open stall, where a red means the stall is currently occupied by another vehicle. This is not the complete functionality of the system, but rather a level of abstraction for the consumer; the data that is measured/collected, will be distributed from device to device via a wireless network, then sent through a gateway where it will be received at The Den as a means of data collection for possible further research.

Due to COVID-19, the original plan was unable to be completed. The contingency plan implemented the sensing in one of the team member’s garages and a limited wireless network. The data was then successfully received over the internet at another member’s house and the simulation provided real time updating of the garage.

# Background

Sponsor Motivation

Although somewhat similar to last year’s “Park My Ride” project, this project is not a continuation of that, but rather a different approach to parking indication. The emphasis from our sponsor, Dr. John Shovic, is on small-scale, low-power and low-cost/maintenance aspects of the project. Outfitting an entire garage with cameras in every stall would be extremely costly, and require a lot of power. This approach is designed to be a quick and cheap way to indicate the status of a parking stall to a commuter who is looking for a place to park within a specific parking garage.

Needs and Stakeholders

The various requirements of this project aim to motivate and inspire the further use of low-cost sensors, mesh networking and simulation use, as part of Dr. Shovic’s future Internet-Of-Things plans. This project was aimed more towards the ability to showcase bigger systems using less prototyping and less money. Rather than make PCBs and hundreds of units to outfit the entire garage, 5 well-made prototypes can convey the same information with proper data seeding to the simulation. Currently, the U of I faculty, primarily Dr. Shovic, are the stakeholders. The units never had the chance to be implemented in the parking garage, but will hopefully be installed by a future capstone group who can expand on the testing and features that we were unable to test and fulfil due to Covid-19.

# Problem Definition

Park-IT-CdA is a parking garage monitoring system located in Coeur d’Alene, Idaho at the parking garage between 3rd and 4th Avenue and along Coeur d’Alene Avenue. It is capable of detecting when a car is parked in a stall and will relay this information to a server which will monitor the parking garage’s statistics such as: time from entering the garage to finding a stall, average time a stall is occupied, average number of stalls used and the current state of all stalls.

The GSS is comprised of 5 Garage Sensor Units (GSU). The GSUs will decide among themselves which is to be the Garage Sensor Master (GSM).

The deliverables are to be the following:

* 5 GSU’s which includes the enclosure, computer hardware – sensors, LEDs, embedded system.
* 1 Garage Sensor Gateway (GSG) to be installed on the roof of the Innovation Den.
* An archive of the software to run the GSS – Arduino sketches, simulation.
* User manual on how to operate the GSUs and simulation
* The portfolio including all documentation of the requirements, design process, project learning, communications, design solution and references.
* All other documents produced throughout the two semesters.

Constraints:

* $1500 budget
* The GSU must be battery powered and last at least a year without recharging
* The communication between GSUs must be wireless
* Communication must be capable of penetrating concrete and brick walls
* The GSU must be within 4”x4”x4”

# Project Plan

## Team Members

Nikolai Tiong

Roles

* Team Leader
* Mesh network design
* Mesh network testing
* Documentation

Zane Goodrich

Roles

* Sensors for the GSUs
* Hardware design – sensors, power
* Hardware testing
* Documentation

Tyrel Parker

Roles

* Processing received data from GSS
* Simulation Software
* Hardware purchasing
* Documentation

Joel Berain

Joined Spring 2020

Roles

* Organizing the Wiki page
* Getting caught up on the project
* Documentation
* Encryption of messages

# Concepts Considered

## Wireless Transmission

The distance between the garage and the Innovation Den is approximately 400 ft. There is Line of Sight between the second and third floors of the garage to the Den’s rooftop where the GSG will be mounted. The garage is constructed out of concrete which greatly reduces the signal strength of wireless signals. The Den is constructed of out brick which does the same. There is also the power requirement which means the wireless technology used must use low power while still covering the distance required.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Range - Urban | Power Usage (Transmitting) | Frequency | Notes |
| WiFi | 50 m | 2-20 W | 2.4 GHz |  |
| Sigfox | 3-10 Km | 158 mW | 900 MHz | Requires SIM card  Service fee |
| Zigbee | 100 m | 100 mW | 868, 915 MHz, 2.4 GHz | Capable of forming mesh network |
| LoRa | 2-3 Km | 100 mW | 433, 868 – 915 MHz | Capable of forming mesh network |

WiFi has too short of a range and too much power usage.

Sigfox requiring a subscription fee ruled that out.

Zigbee has too short of a range.

## Microcontroller

A microcontroller will be needed as the base hardware due to their low power consumption, low cost and small size.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Current Usage (Sleep) | Processor | RAM | Storage | I/O Pins | Wireless Module |
| Arduino Uno | 30 mA | AVR 16 MHz | 2 KB | 32 KB | 20 | Separate |
| Arduino Pro Mini | 4.2 uA | AVR 16 MHz | 2 KB | 32 KB | 20 | Separate |
| Arduino MKR WAN 1310 | 104 uA | SAMD21 48 MHz | 32 KB | 256 KB | 20 | LoRa |
| ESP8266 | 20 uA | L106 80 MHz | 32 KB | 512 KB | 16 | WiFi |
| Raspberry Pi | 100 mA | ARM 1.5 GHz | 4 GB | Variable >1 GB | 40 | WiFi |
| Adafruit Feather M0 | 300 uA | SAMD21 48 MHz | 32 KB | 256 KB | 20 | LoRa |

The Raspberry PI and Arduino Uno use too much power to be usable.

Everyone in the group has taken CS443 and is familiar with using the Arduino Uno. We purchased some Arduino Pro Minis first and tested them out. We had mixed results with them with being unable to power them on and upload sketches.

The next board tested was the MKR WAN 1310 which had a built in LoRa board. At this point we had decided on using LoRa with the Radiohead Library. However, upon testing it, it was revealed that the LoRa radio on this board does not support Radiohead.

## Battery and Power

The battery needed for fit in the 4”x4”x4” enclosure and leave enough space for the other hardware.

Concepts considered:

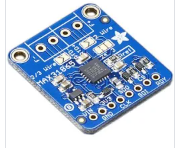
* Combining AA or D batteries in series and parallel
* 3.7 V LiPo battery packs
* Purchasing 18650 Li Ion batteries and creating our own battery packs
* Adding solar panels to supplement the battery
* Putting the microcontroller to sleep always when not in use and only waking on a timer or interrupt

A battery of 10,000 mAh capacity was calculated as the minimum needed. See the appendix for assumptions, parameters and the calculations.

Combining batteries to create a battery pack involves too much risk with internal resistances and different voltages for each individual battery possibly causing issues with discharging and were ruled out.

**Alternate sensor/detection methods considered**

VL6180X – Adafruit time of flight sensor:



This sensor is similar in functionality to the Ultrasonic sensor that we ultimately used. A laser is used to ping and read, measure the time between the two, and calculate the distance to the object in front of it. The voltage is appealing for a 3v3 system, being only 2.7v to power the module.

Potential issues:

Cost: these sensors are roughly $14-23 dollars per unit, which significantly reduces our budget if they fail to work as we intended them to.

I2C interface:

Although there isn’t usually an issue with multiple modules using the I2C bus, in fact, this is the specialty of this medium, allowing for communication to dozens of addresses with only 2 bus wires, one to send and one to receive. However, when working in and out of deep sleep modes, the less time that is spent within an interrupt routine the better, as well as messing with global variables and having to implement semaphores and mutexes, to keep from freezing, corrupting or crashing the system. It seemed more reasonable to use a sensor with wiring independent of other modules in the system.

KY0030 microphone sensor:



This sensor can be used to detect the volume of surrounding noise. We figured that tuning it to recognize when a vehicle was in front of it, based on raised levels in the unit’s sound threshold, that we could have a cheap and accurate way of detection. The units are very cheap and require very little power to keep them running.

Issues:

Tyrel tested a couple of these sensors at his house early in the first semester, and quickly discovered that these sensors are very sensitive to the noises they trigger to. A snap of the fingers would set them off at the lowest sensitivity setting, making it impossible to use with any sort of accuracy in a noisy garage, looking for noisy vehicles.

LLC-161 light sensing resistor, paired with a 650-NM laser module



Another one of our considerations in vehicle sensing was a “broken light” sensor. The laser would be positioned at an angle, and project from the upper side of the stall, to the opposite and lower side. On the receiving end of the laser would be the light sensor, if the sensor detected light, there was no vehicle in the way, and thus the stall would be open, and if it no longer detected the light source, that meant that a vehicle was blocking it, and the stall was closed.

Issues:

Although being one of the cheapest sensor ideas we considered, the slightest bit of wind, dirt or tampering could easily send these units into false reads. Testing from home proved that these units are very picky to the angle at which the light is received; a few degrees off would “break” the light source and the system would detect a vehicle when only the laser had moved a tiny bit.

E0S2-1200TVL Micro FPV Camera:



Suggested to us by various faculty members, was the use of a low-power camera module to detect vehicles. This would work in much the same way as the Infrared sensor, and would compare frames to detect when there was a change, or a vehicle present.

Issues:

The main issue with using a camera, or at least the issue with our particular use, is the power usage. To run our Units on battery life for a year, there is no way a camera could be used for even a few minutes per day, let alone all the time, or even on interrupts. While this would be a great option in a “wired in” system with unlimited power resources, it was simply not feasible for our project.

## Enclosure

## Simulation

## Encryption

Data encryption will be implemented for data packets being sent and received across the network. Encryption is used for data security. There are two main purposes to consider for data security across our network:

* **Data Concealment:** Data that it is flowing throughout the network could be intercepted by an unwanted entity. To protect this data, encryption can be implemented to scramble and hide the “actual” data that is being sent. This will increase data security over our network.
* **Data Authorization:** When receiving data across our network, a data packet sent by an unwanted entity could be intercepted. To help avoid this problem, data packets could be “stamped” with an indicator that shows it is an authorized packet for our network. Data being sent can be configured in a way that when it is received it can be first checked for authorization before it is interpreted. This will not only help secure our network, but also avoid waisted time and resources on data that was not authorized by our system.

Many existing encryption libraries can be found that can be implemented into our network. We could use one of those or create our own encryption class. If we create our own, there are also many encryption methods to consider (e.g. Private or Public key).

# Concept Selection

## Wireless Transmission

Of the 4 options considered, LoRa fit the requirements the best. It was capable of the range we needed and used low power. It also made use of the Radiohead library that Dr Shovic suggested that we consider using. We purchased some Adafruit LoRa radio modules and tested them out on Arduino Unos and Megas. This was successful when taken out on a field test between the garage and the Innovation Den, as well as within the garage. Nikolai also found a mesh network implementation that could be used as a base to build upon.

## Microcontroller

The Adafruit Feather M0 was selected as the microcontroller after the plan to use the MKR WAN 1310 fell through. However, this didn’t happen until January 2020 which left little time to obtain and test another board.

## Battery

Due to the difficulty in sourcing a large enough battery, the battery decided upon was a 2500 mAh capacity one. The uptime of the GSU would be timed and then extrapolated to determine how large a battery would be needed to last a year.

Adafruit has multiple components that can be combined to provide solar charging. They have 1W 6V solar panels, the charging unit with regulator, JST connectors and battery packs. The solar panel will not provide much power since it will not be in direct sunlight, but is something can be added relatively cheaply: $40 per GSU and help squeeze out as much battery life as possible.

## Enclosure

## Sensors

The sensors

**HC-SR501 PIR Motion Sensor 3v:**



This sensor is what we chose to “wake” the system. The Arduino board itself goes into a very deep sleep when there is no activity, and shuts down almost everything, waiting for an external interrupt to trigger and wake it up. This sensor has the ability to wake the Arduino in this way.

The module itself looks for heat signatures, and compares them to its previous reading. In this case, if the sensor sees no heat, it just waits for a few milliseconds and checks again. It does this at an extremely low rate of power consumption. When it finally detect that a heat signature has changed, or it “sees” heat, it sets its data pin to HIGH, and tells the Arduino to wake up.

**Factors in selection**

Power

This unit runs at 3v power as well as logic, and uses less than .1Ma when it is idle.

Price

When purchased in ordered of 5 or more, this unit is less than $1.50

Form factor and ease

Having only three pins, a power, ground and data line, this unit is extremely reliable and easy to work with. Also, its 3v version is much smaller and more linear shaped than the 5v counterparts, making it easier to place in a small enclosure.

Testing

***See testing results page in final report***

**HC-SR04 Ultrasonic Distance Sensor**



This sensor is used in conjunction with the PIR sensor above. It pings, or sends a very brief sound wave out from one side, and receives the rebounded ping on the other. The sound bounces off an object and based on the time it takes to come back, the sensor can determine the distance that the object is from itself. This sensor is used to redundantly check the presence of a vehicle after the PIR sensor has woken the system It performs 3 checks rapidly, and all three must be a reasonable distance reading, or the values are rejected and retested. This is done to avoid changing the LED to a false state based on false readings. The 2 – to – 1 vote ensures that erroneous reads are not accepted.

**Factors in selection**

Price

These units are one of the cheapest and most available sensors there are. One unit costs less than $2, and this price drops even more when purchased in small quantities.

Power

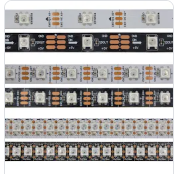
These units use less than 2mA statically, and they are on for less than a second whenever the system is woken from deep sleep. They work at both 5v and 3v power as well as logic.

Availability

These units are extremely easy to obtain, and can be purchased from a multitude of different vendors, making pricing and shipping much more easy and affordable.

**The LEDS**

WS2812B RGB LED Strip (One single LED)



For the LED indication, we used one single LED from these strips. The LED is used to signal to the commuter the state of the parking stall. As they are flat, they don’t emit light very wide, but rather pinpoint and harsh. A custom diffuser was made to enclose the LEDs and spread the light, making them visible in all lighting conditions.

**Factors in selection**

Price

Most Arduino hobbyists have a least one or two of these long LED strips laying around. The cost of one LED from them is virtually negligible. If we were outfitting multiple Units at once, a single $20 strip of these LEDs could cover over 100 units, at about 20 cents per unit, making it our cheapest component in the GSU

Ease of use

These LEDs are extremely easy to work with. I have personally written several routines and a couple of libraries for controlling them in personal projects. They are only one way in their communication. The microcontroller fires a status to them and moves on. For this reason, there are only 3 pins just like the PIR sensor, and the form factor is very small.

Visibility

These little LEDs are much brighter than the standard bulb-shaped ones that are found in many hobby kits for Arduino programming. The light is very strong and vibrant, and can be easily manipulated with various diffusing techniques.

**Other Modules**

The other modules and components used for the final unit design have not changed throughout this project, and are very standard, not subject to various operational traits and functionalities. They are simply listed below, as their properties and uses are standard for any use:

* 2500Mah Lipo battery
* Lipo charging board
* Latching power switch
* Real time clock
* LoRa antenna

## Simulation

## Encryption

Our very own encryption class will be created to provide data security across our network. It will implement a private-key method to conceal the data, and it will add a data message at the beginning of each data packet to provide data authorization.

To fulfill each purpose of our encryption class the following will be implemented:

* **Data Concealment:** At this stage of development, the data flowing through our network is not considered “sensitive”. If data from the network is read by an unwanted entity, it would not be considered a high-risk security threat. Efficiency of data interpretation is probably more important to our network at this time. Implementing a simple, yet sophisticated, encryption method should suffice.

The encryption class will consist of specific multiple private keys. It will have a predetermined pattern that instructs which key to use at each stage of the encryption/decryption process. The keys can be changed internally if needed.

* **Data Authorization:** For data authorization, a simple phrase will be chosen to add at the beginning of each data packet. Each message being sent across the network will be tagged with this phrase, so each message being received will recognize that data packet as an authorized message.

# System Architecture

**Design**

We wanted the design of this project to reflect the footprint of the power and sensors used within it. The enclosure is very compact, the sensors are tiny and use very little power. The mesh networking protocols are speedy, reliable and small as well. The simulation accompanies these other properties of the project by being hosted on a webpage that runs on a board not too unlike the one that is powering the units themselves.

The whole project, although implemented with full power, and a house-to-house communication with WiFi in between the LoRa transmissions, was designed to be deployed in a parking garage, and last as long as possible on a single battery charge. Moving forward, the programs running on the units can be changed to match the garage setting, as well as the networking and simulation. Only minor changes need to be executed to achieve its original and intended functionality.

**System Functionality**

The components and their basic system function can be found above in the Concept Selection area, and a more detailed explanation of system-wide communication and functionality can be found on our Github:

Below are the links to the various data flow diagrams of the full system architecture on each level, as well as a Fritzing diagram of all the components and how they are wired together:

Context Diagrams:

Level 0 - <https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/PIC%20context%20diagram%20level%200.pdf>

Level 1 - <https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/PIC%20context%20diagram%20level%201.pdf>

Level 2 - <https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/PIC%20context%20diagram%20level%202.pdf>

Fritzing Diagram - <https://github.com/Jaxal83/parkingSensor/blob/master/Spring%202020/Hardware%20Design/Park-It-CDA%20fritzing.JPG>

**Novel Features**

Enclosure

The enclosure is the result of many changes to the project throughout the last two semesters. Every time a component changed; the enclosure changed. Zane designed the enclosure while learning Fusion360 modeling through Udemy.com, so each iteration was a litter more suitable and acceptable than the last. The earlier models were very rough, and did not have any inner cavities or standoffs to accommodate the various components. They were zip-tied and hot-glued to begin with and were ultimately measured and designed around in the final build.

Simulation

The graphical representation of the simulation is a very easy to read layout of a parking garage. Tyrel made the simulation small and concise so that the speed and availability of the webpage are consistent with the rest of the project. The entire webpage is hosted on a small microcontroller as well.

The overall setup of the project

The biggest “us” factor that is present in this project is the way the data is flowing for the demonstration. We had to pivot very hard due to the Covid-19 circumstances and were able to achieve house-to-house communication and a way to test our various portions of the project. This can be found in much greater detail in our Github.

**Satisfaction of requirements**

Sensing vehicles

As mentioned above, the two sensors in our units work in conjunction to accurately detect when a vehicle is present below. Erroneous results are thrown out, and the LED only changes when the stall status changes. A complete demo video of this can be found in the Expo presentation on Github

Mesh networking

The units are able to communicate with each other, as well as the gateway. Units that are too far from the gateway can send their status to the nearest neighboring unit, and so on until a unit close enough to the gateway is able to send the status of the stall.

Simulation

The simulation shows an accurate representation of the parking garage, as well as the parking stalls in Zane’s driveway. The gateway receives a LoRa transmission and is able to send it via WiFi to the Raspberry Pi that is feeding the data to the webpage. This happens very quickly and stall indication via the webpage can be easily and quickly portrayed. A demonstration of this can also be found in the Expo presentation section of our Github.

# Design Evaluation

## Vehicle Detection and Indication

The unit needs to be in an active traffic/parking zone and monitored to verify the following:

The LED indicator will blink Green periodically, to indicate that the stall is empty. It will need to do this while the rest of the system is asleep. Upon the detection of a heat signature (vehicle) it will wake, triple-verify that the vehicle has entered the space, and change the indication color from Green to Red. It will then need to go back to sleep and blink Red periodically, while the system is asleep, and until there is a change in the stall.

Various vehicles will need to be verified to ensure that all types are detected, and that the radius of the PIR as well as the range of both the PIR and Ultrasonic sensors are catching any vehicle, while also not updating falsely due to the presence of a vehicle that is not within the lines of the stall in which this particular unit operates.

## LED Indication

The LED always needs to be visible. This means testing its visibility during the morning, afternoon and night. The LEDs on each of the units should not drift so much as to cause a strobing effect in the garage.

## Hardware Enclosure

The enclosure for the parking unit should be tested on a similar surface to that of the parking garage ceiling, with different strengths of mounting tape, to ensure that it holds indefinitely, without the risk of damaging a vehicle or damage to the surface on which it is mounted. It will need to be placed in a test area and left for enough time to ensure that it is still holding strong and will continue to do so.

## Mesh Network

These are the milestones and things that need to be tested on the Mesh Network to conclude that it is performing as expected.

GSUs can transmit data to the GSM either directly or via another GSU. GSUs wake up on a timer to transmit, then go back to sleep. The GSUs and GSM need to be in sync with one another to send and receive data. The receiver is only going to be active for a short period since it uses ~20mA while active.

The GSM can transmit data to the GSG at the Den. This needs to be done on a periodic basis to clear the memory of the GSM since we don’t have a lot of memory to work with.

Data transmitted is encrypted – the payload needs to be encrypted since anyone with a LoRa radio would be able to receive the packets and decipher them. Especially since some of the messages will be control signals.

## Data Collection

The GSUs need to be able to store the sensor data and a timestamp. The GSM will transmit data to the GSG as it is received.

The server in the Den will permanently store all data that comes in.

Data will also need to be collected to calibrate the vehicle detection software. This means recording the average time that a commuter takes to park, to leave and the duration in which their vehicle is occupying a stall.

## Data Analysis

The Simulation must take in this data to analyze and so that it can work out average time from car entry to carpark occupancy.

## Testing Results

The final COVID-19 affected product produced the following results:

Sensing Vehicles

* The GSU was assembled and an enclosure created
* The GSU was tested by placing it above the ground in Zane’s garage and connected to mains power
* The GSU indicated a green LED to show that the space was open
* The sensing was tested by driving his truck forward into the garage, once the vehicle was under the sensor, the LED changed to green
* After reversing the truck, the LED changed back to green

Networking

* The GSU created a message to send every 10 seconds
* The message used the format that was originally created (this was redundant for this test however). This consisted of the <1>, GSG ID, GSM ID, GSU ID, parking stall status and the date timestamp in UNIX time. The <1> is an identifier to let the GSG know the message is to be forwarded to the MQTT server.
* The Dragino GSG received the sent message and forwarded the message onto the MQTT server via the internet at Tyrel’s house

Simulation

* The MSTT server receives the GSU message and parses it
* It then sends the data to the simulation software
* The simulation then updates the stall status
* The status of the parking garage can be viewed in real time via a webpage

Given the circumstances, the testing results are considered a success: the key requirements of sensing vehicles and indicating whether a car was in a stall, as well as displaying a simulation via a web browser was successful. The testing to get LoRa communication between Nikolai and Zane’s house was unsuccessful, however, this range requirement far exceeded the original requirement of 400 ft.

# Future Work

Due to COVID-19, the original design was not implemented. Many features were missing from the final test used for the Expo presentation.

Features Implemented

* Sensing a vehicle using ultrasonic and PIR sensors
* LED indication of parking stall status
* Point-to-point communication over LoRa – 2x GSU to GSG
* Addressing scheme over LoRa
* LoRa message format
* GSG to MQTT communication over internet
* Simulation
* Real Time Clock for timestamps

Features Not Implemented

* Battery requirement was removed and never tested
* Sleep and power usage reduction
* Solar panel and charger were not purchased
* Encryption
* Mesh network – changed at start of Spring 2020 to a star topology network
* Star network – long range communication of 1.25 miles was inconsistent and point-to-point was used

A future group project should work on the following:

* Build the star network with multiple GSUs communicating with the GSM
* Synchronize the GSUs so that there isn’t more than one transmitting to the GSM at a time
* Implement the sleep and waking up from an interrupt from the DS3231 RTC
* Implement the encryption module
* Reduce power usage
* Revamp the GSU enclosure and add the solar panel
* Modify the simulation to reflect the garage’s layout
* Source a battery

# Appendices

**Use case diagram**

<https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/capstone%20use%20case.jpg>

**Context diagram level 0**

<https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/PIC%20context%20diagram%20level%200.pdf>

**Context diagram level 1**

<https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/PIC%20context%20diagram%20level%201.pdf>

**Context diagram level 2**

<https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/PIC%20context%20diagram%20level%202.pdf>

**Power usage**

<https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/Power%20Usage.docx>

**Battery usage**

<https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/battery%20usage.xlsx>

**Original GSU 3D Model Concept Design**

<https://github.com/Jaxal83/parkingSensor/blob/master/Spring%202020/Hardware%20Design/Concept%20Design%20GSU.png>

**Final 3D Model of GSU**

**(front)** <https://github.com/Jaxal83/parkingSensor/blob/master/Spring%202020/Hardware%20Design/final%20model%20front.PNG>

**(rear)**

<https://github.com/Jaxal83/parkingSensor/blob/master/Spring%202020/Hardware%20Design/Final%20Model.PNG>

**Finished Assembly of GSU**

<https://github.com/Jaxal83/parkingSensor/blob/master/Spring%202020/Hardware%20Design/Final%20Model.PNG>

**Schedules:**

**9-Month calendar**

<https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/9mo%20calendar.docx>

**Master gantt**

<https://github.com/Jaxal83/parkingSensor/blob/master/Fall%202019/Master_Gantt.ods>

**Programs, Code, Sketches**

**Arduino Sketches**

<https://github.com/Jaxal83/parkingSensor/tree/master/Programs/Arduino%20Sketches>

**Libraries**

<https://github.com/Jaxal83/parkingSensor/tree/master/Programs/Arduino%20Sketches/Libraries>

**Component Documentation**

**Manuals and Datasheets**

<https://github.com/Jaxal83/parkingSensor/tree/master/Programs/Manuals%20and%20Datasheets>

**List of Hardware**

<https://github.com/Jaxal83/parkingSensor/blob/master/Programs/Hardware%20List.docx>

## Battery Power Calculations

I create a spreadsheet with the different types of ways that the battery is going to be drained in the GSU. I then do some estimates as to the power usage based on datasheets and how often each will be used in a day.

The following assumptions were used:

* 10,000 mAh battery
* 1% battery drain per month due to self-discharge
* Transmission over LoRa will be at 50% power – 50mA
* Receiving over LoRa uses 20mA, there will be a 20 second receive window when a GSU is receiving
* A transmission from GSUs to the GSM will be every 30 minutes between 6am and 8pm
* Sensors activate on interrupt and will run for 10 seconds at a time
* The PIR sensor activates when a car parks/motion is detected
* The ultrasonic sensor will be used when a car is parked every 30 minutes afterwards
* A 10% duty cycle for the LED will be used, which corresponds to 10% of a 30mA LED
* The Arduino will be using current when active
* There will be a constant current drain when in sleep mode
* Solar charging is currently unknown and will not be included yet

Parameters:

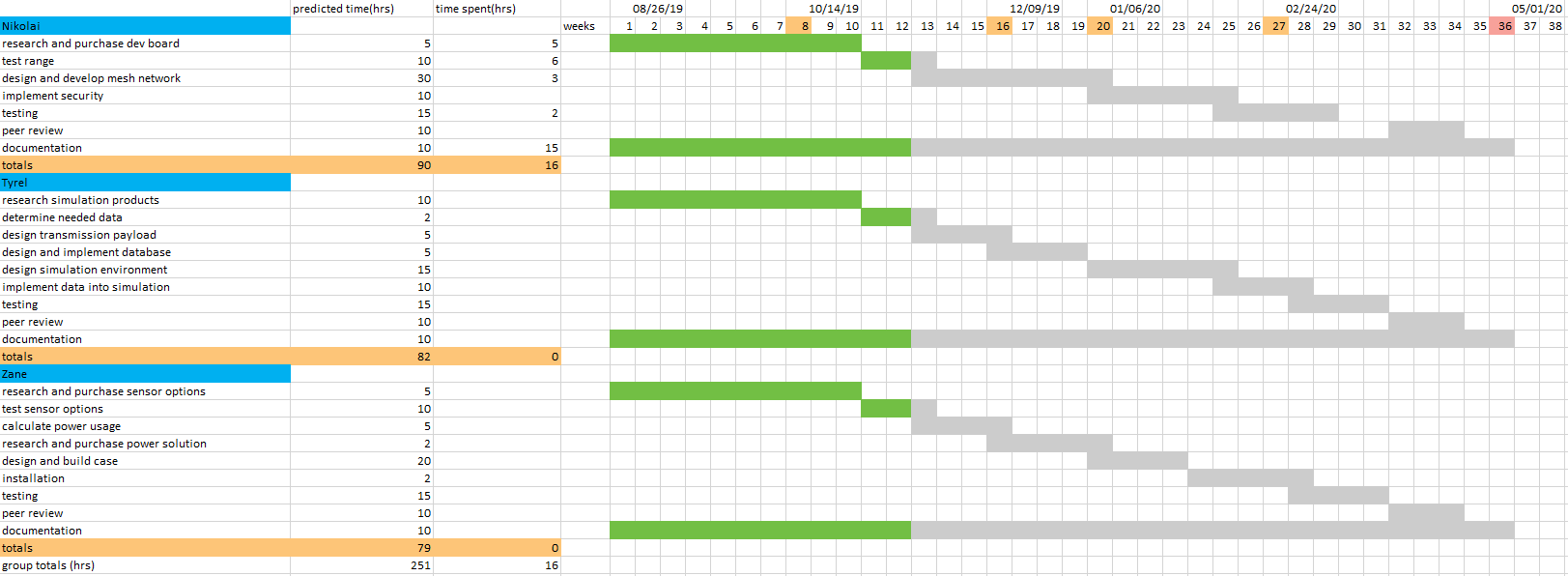
* Days per year 365

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Current Used (mA) | Duration (s) | mAh per Day | mAh per Month | mAh per Year |
| Transmitting | 50 | 1 | 0.39 | 11.67 | 141.94 |
| Receiving | 20 | 30 | 4.67 | 140.00 | 1703.33 |
| PIR | 2 | 10 | 0.16 | 4.67 | 56.78 |
| Ultrasonic | 6 | 10 | 0.47 | 14.00 | 170.33 |
| LED | 20 | 0.05 | 2.00 | 60.00 | 730.00 |
| Arduino (Blink LED) | 30 | 0.05 | 3.00 | 90.00 | 1095.00 |
| Arduino (LoRa, Receiving) | 30 | 30 | 7.00 | 210.00 | 2555.00 |
|  |  |  |  | Total Used (mAh) | 6452.39 |
|  |  |  |  | Remaining (mAh) | 2347.61 |
|  |  |  |  | Idle Draw (mA) | 0.27 |

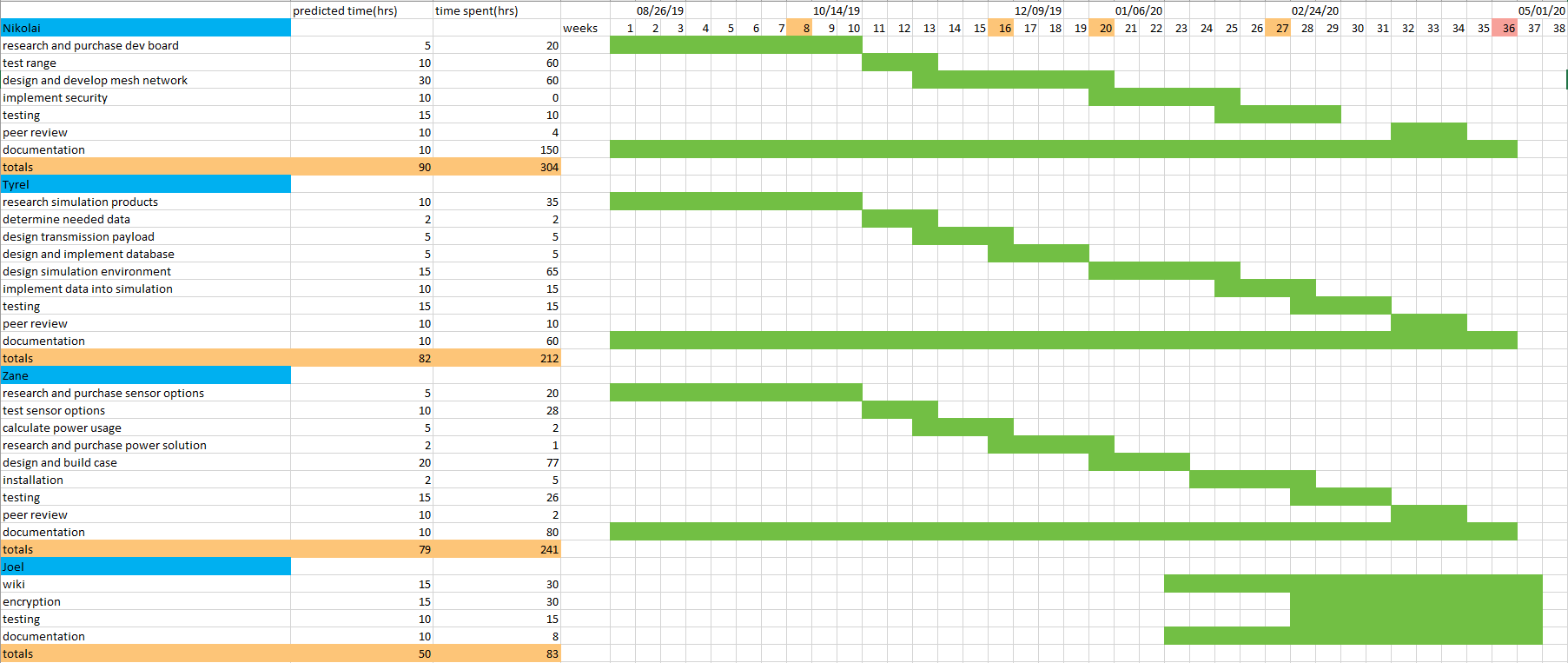
* Frequency per hour 2 (LoRa and sensor)
* Frequency per hour 3600 (LED)
* Hours active per day 14
* Days per month 30
* Available capacity on a 10,000mAh battery is 8,800mAh after accounting for the self-discharge

We need to aim for an idle current draw of 270uA in order to last for a year before we consider solar charging.

## Original Schedule



## Final Schedule



## Shared Folder Structure

Client Meetings

Documentation of meetings with the client: Dr Shovic.

Fall 2019

All documentation, submitted to BBLearn or otherwise created, from the Fall semester. Includes the project portfolio submitted in February 2020 documenting the first semester.

Programs

Contains all the Arduino sketches, libraries and simulation software created during the project. Includes instructions on how to operate the hardware and software.

Spring 2020

All documentation, submitted to BBLearn or otherwise created, from the Spring semester.

Team Meetings

Documentation of all team meetings.