# Secure, Anti-Theft, Flexible, Engineered Taillight — SAFE T

*App-Controlled Taillight with Turn Signals for Bicyclists with Theft-detection*

Department of Electrical and Computer Engineering
EEL 4915 – Senior Design 2
Advisor: Dr. Samuel Richie
April 26, 2022

Group 34

Jonathan Diaz (EE)
Jay Kurczy (EE)
Brandon Therrien (CpE)
Paul Wilson (CpE)

# Contents

# List of Tables

# List of Figures

# Executive Summary

Before the motor was invented, simple mechanical or biological means of transportation were very prevalent. The most traditional method of transportation for thousands of years has been horses, with the carriage or cart coming about quite some time after. It was not until 1817 that the bicycle was invented. With the advent of realization of impact in regards to Climate Change, green transportation has become an even hotter topic. Bicycles in particular are a classic mode of transportation, with $<1\%$ to $1\%$ of workers commuting via their favorite two-wheeled vehicle. This option is much more viable due to its mechanical operation.

Our motivation for creating what amounts to an add-on pack for bicycles stems from the high incidence rates caused by motor vehicles sharing the road. Due to the large size of motor vehicles and the speeds which they operate at, the safety of bicyclists is compromised, this only gets worse as cars capable of faster speeds and quicker acceleration come out every year. This is even further exacerbated by intoxicated driving, drowsiness, and simple lack of attention being paid to the road.

We will be creating a product that provides lighting and signals as its primary purpose. The product will also additionally feature anti-theft systems, including an alarm that will sound when the bicycle's motion is detected. The signals will consist of a left and right turn signal to indicate to automobile drivers what the bicyclist's intent is. Further lighting systems are provided to constantly illuminate the bicyclist in order to prevent automobile-related injuries, this includes a rear brake/tail light which will be red along with a front headlight that will illuminate the path in front of the rider.

Control and interaction of/with the device will be established through buttons and a Bluetooth-enabled software application available on iOS and Android which are the largest mobile platforms. Buttons allow activation of the turn signals and lights along with changing the brightness, toggling of Bluetooth, and a switch to toggle the alarm. All functions found on the application will have a physical implementation. Our hope is to provide an application that provides a condensed control box which the user may more easily interact with, even allowing interaction from a distance to provide easier arming and disarming of the alarm and other anti-theft features.

Relevant standards of Bluetooth and IPXX will be evaluated and compared in regard to their efficacy. We expect our customers to be able to merrily traverse their normal trails, rain or snow, without our product failing. Waterproofing will be a major concern for many of the components including the MCU, Alarm, circuitry, etc. Other standards for evaluation include Battery life, charging, etc. Alarm sound and decibel level will be important for battery life and theft prevention. Bicycle handlebar size is vital for our physical design. Most importantly, the lighting shall be examined for a high luminosity, visibility is only improved with light while battery life is decreased. Further information will be included as necessary to compensate for unanticipated standards.

Installation of the product shall be such that the user may easily strap on the signal buttons/indicators and affix the main unit which will contain the circuitry. The product will be easily removable to allow for ease of transporting, charging, and to allow it to be affixed to another bicycle.

This product is designed to be a low-cost system, this is done to increase market reach. This product does not contain many expensive parts and should be fairly simple to develop. The product application will also be fairly simple and will be available on both major platforms for further reach. With these things in mind, the everyday user should be able to easily purchase our product, download and install our application, install the product, and get started very quickly and with very little downtime. The difficulty on the part of the user is kept to a minimum. For a minimal amount of effort, we may increase the safety of the user immensely.

This product combines a physical hardware aspect with a software application aspect and as such will require a lot of communication between systems. Our research will be significant, especially with regards to the unfamiliar, but is within our capabilities as Computer and Electrical engineers. This document will contain our research, decisions, and notes.

# 1 Introduction

## 1.1 Background

Modern-day life is defined by the advent of ever-miniaturizing technological processes, allowing for the implementation of circuitry in every facet of our existence. The prevalence of electronic products has increased exponentially, allowing for inventors to boldly explore possibilities that were mere science-fiction only some few decades ago. Problems that could not be solved by conventional mechanical means now have endless solutions implementable by microscopic circuitry, complemented by complex software. It is in light of this that we have pursued our degree paths and now must design a functioning product.

## 1.2 Purpose

This report serves the need to document and explain our design choices and the culminating project which is the goal of this class. It will contain all relevant information, including but not limited to the testing of the product and its performance therein, along with schematics and developmental iterations or prototypes. Further graphs, charts, and depictions of products will also be included in order to more easily explain and complement ideas and texts. Relevant background information, including an explanation of standards, comparison to competitors, along with budgeting and financing shall also be included.

## 1.3 Market

Technological advancements in automobile systems have created a higher risk environment, not only for those driving a motor vehicle but also pedestrians and bicyclists. Accessibility to higher speeds has increased along with lowering acceleration times, these combine to create a greater effect. Motor vehicles came about in the late 19th and early 20th century, namely in 1908 with the Model T for the USA, yet national standards were not established until the mid-1960s. While safety regulations have lowered the death toll, greater accessibility to motor vehicles and higher speed-capability along with the ever-present intoxicated driving leave non-motorist safety still in flux. According to the National Highway Traffic Safety Administration (NHTSA) 2015 statistics, there were over 800 fatalities and approximately 45000 injuries to cyclists. Further, the NHTSA reports that 28 people die everyday from drunk driving.

In Florida, cyclists are required under statutes 316.155 and 316.156 to signal their intention yet hand signaling is not very common. News4Jax reported in 2019 with respect to NHTSA's released statistics that Florida led the nation with 125 of the 783 bicyclist fatalities. Though many steps have been taken to ensure the safety of bicyclists there is still a gap which this team seeks to rectify. Specifically, lighting is currently unregulated even though it is shown that more accidents

happen at night and those without lights are more likely to be in an accident. This goes back to our goal of increasing bicyclists' lighting in order to make them safer and make them more easily distinguishable on the road, even in harsh weather conditions.

In Figure 1.3.1 we see that Fatalities are generally not the fault of the bicyclist,as a large portion are killed by being rear-ended. Two percent of fatalities are due to the Cyclist failing to yield, so this really shows that automobile drivers are the main issue.

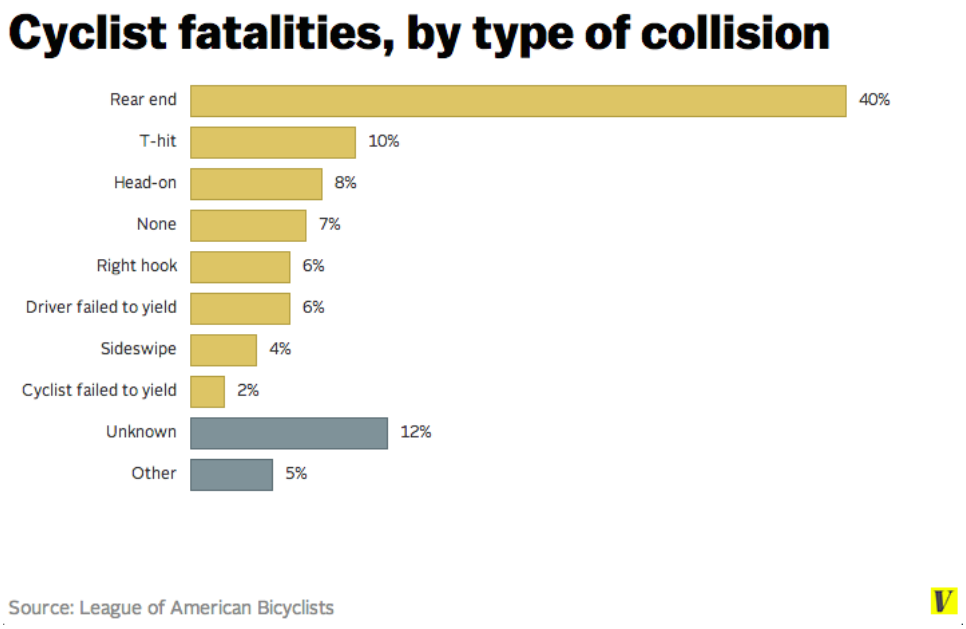



Figure 1.3.1: Cyclist fatalities, by type of collision [1]

In Figure 1.3.2 we further confirm that automobile drivers are the primary issue when concerned with bicyclist fatalities with a large majority being caused by hit and runs, and careless or intoxicated driving. Cyclist visibility helps to greatly reduce the chance a driver will hit them but it does not protect from intoxication or carelessness.

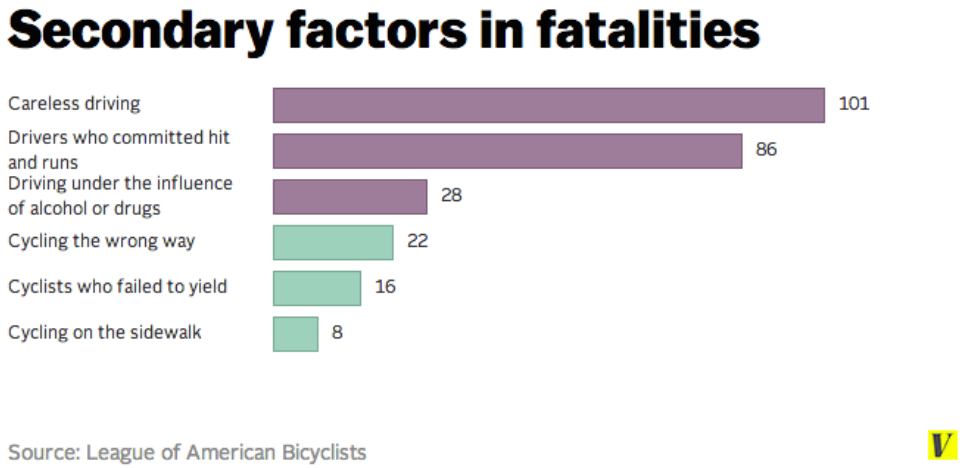



Figure 1.3.2: Secondary factors in fatalities [1]

Figure 1.3.3 demonstrates that—in line with the previous figures—bicyclists' own safety through use of a helmet does not affect their chances of being involved in a motor vehicle accident.

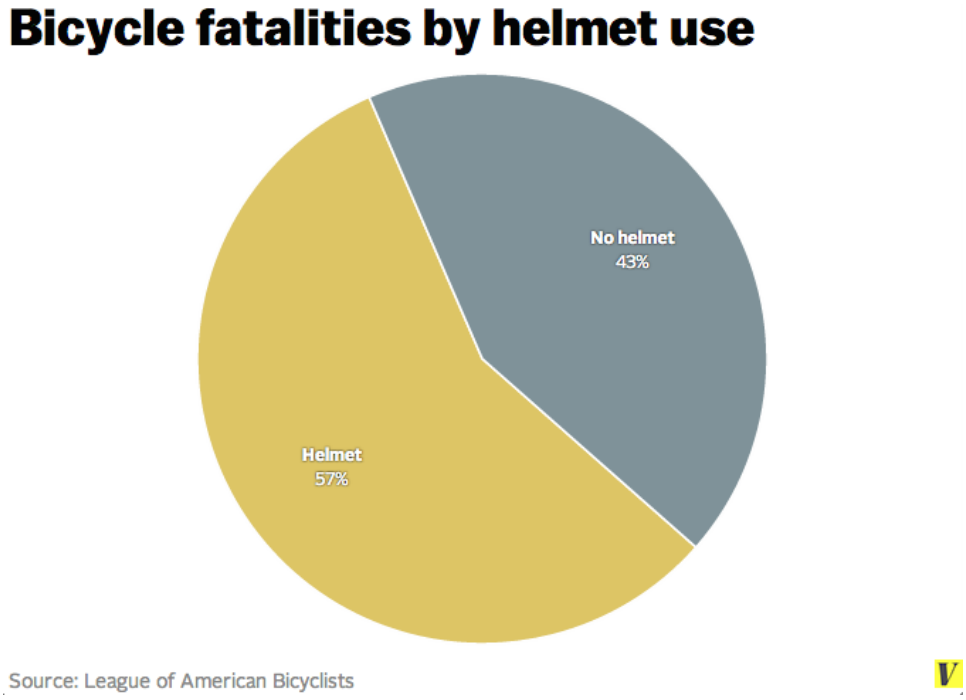



Figure 1.3.3: Bicycle fatalities by helmet use [1]

Figure 1.3.4 demonstrates that the highest concentration of bicycle fatalities occurs within urban areas, where concentration of vehicles (both automobiles and bicycles) is at its highest. These areas do tend to be more well-lit than their Rural counterparts but still tend to have insufficient nighttime illumination to prevent pedestrian and bicyclist involved collisions.

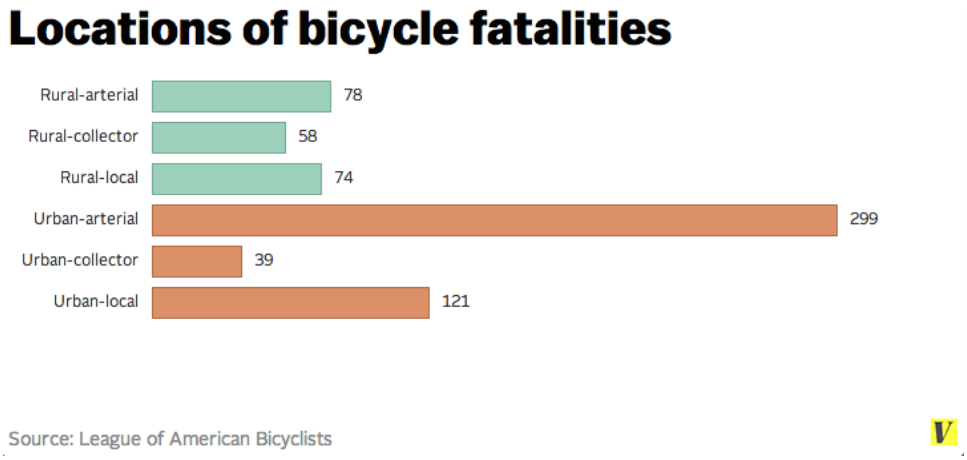



Figure 1.3.4: Locations of bicycle fatalities [1]

Table 1.3.1 demonstrates that time of day heavily influences the number of bicyclist fatalities with approximately 50% of deaths occurring from 6pm to 6am. Most Deaths happen later in

the day, with 52% occurring after 3pm. Proper lighting affixed to a rider's bicycle should greatly mitigate the effect on vision caused by night time.

Table 1.3.1: Bicyclist Deaths by Time of Day [2]

| Distribution of bicyclist deaths by time of day, 2019 | | |
|---|---|---|
| Time of day | Deaths | % |
| Midnight - 3 a.m. | 58 | 7 |
| 3 a.m. - 6 a.m. | 52 | 6 |
| 6 a.m. - 9 a.m. | 105 | 12 |
| 9 a.m. - noon | 84 | 10 |
| noon - 3 p.m. | 96 | 11 |
| 3 p.m. - 6 p.m. | 130 | 15 |
| 6 p.m. - 9 p.m. | 177 | 21 |
| 9 p.m. - midnight | 139 | 16 |
| Total* | 843 | 100 |
| *Total includes other and/or unknowns | | |

## 1.4 Problem Statement

As aforementioned, this team seeks to rectify some safety issues associated with bicycling. Our product, SAFE T is intended to provide clear turn-signal indication along with head and tail lights to indicate rider position to automobile drivers, plus some extra anti-theft features. The requirements we wish to fulfill are outlined in section 2 and explained in section 4. This is an exhaustive list relating to the lighting, inter-system communication, physical design, theft prevention, sound level, battery, etc. While there are many competitors out there, we hope our product will stand above the rest.

## 1.5 Justification of Project

Due to the statistics depicted above, we believe it is important to service the bicyclist community in order to improve their safety. This product may easily be used as an add-on to any smaller vehicle which does not come with signals, lights, or anti-theft features and thus serves a large market. The increased interest in green transportation also serves to widen market growth and thus our reach. Our main priority will remain the safety of bicyclists, we hope the addition of turn indicators along with head and tail lights will serve to decrease incidents of collision with motor vehicles. Our second priority is to develop anti-theft features that protect the bicycle from

ne'er-do-wells. Heavy consideration will be placed on tamper prevention to keep potential thieves from easily circumventing our impacted systems.

## 1.6 Organization of Report

This report is organized as established in the table of contents and shall contain all relevant information in its respective section. Our members will reflect on their experiences and provide insight into various decisions and design choices. The report that follows is a collaborative effort by all researchers, with sections being written and revised by several members each; that said, it is impossible to specify the sections by group member, and it will be clear that multiple voices present themselves in almost every chapter. The design of the project is similar in that the collaborations of each member become conjoined to form a single, final, unit.

## 1.7 Existing Products

There are many competitors out there however, their products generally serve a signal of our proposed functions. For instance, searching "Bicycle Light" or "Bicycle Turn Signal" on Amazon provides plenty of resulting products which serve the function of headlights or turn signal indicators. Some of the competitor products do offer extended features and ease of use via a simple, small remote. A search of "Bike Antitheft Alarm" reveals a plentitude of attachable alarm systems that work via motion sensor and can be controlled with a remote but none that provide any further features. Some competitors do combine a tail light with an alarm but no reputable dealer offers turn indicators as well.

# 2 Project Objectives and Requirements

## 2.1 In-Scope Objectives

These objectives are those that are within our ability to do so within the allotted time and with our given skillsets. Each objective is an overarching goal we wish to achieve and pertain to their relative product aspect.

Table 2.1.1: In-Scope Software Objectives

| Software |
|---|
| Interface with MCU |
| Remote Control of Lights |
| Slider to Adjust Light Brightness |
| Compatible with Android and iOS |
| Lightweight, easy to install |
| Easy to use, very accessible |

Table 2.1.2: In-Scope Hardware Objectives

| Hardware |
|---|
| Small and portable |
| Waterproof and environment proof |
| Tamper proof |
| Smartphone-enabled |
| Fast and accessible charging |
| Long battery life |
| Front and Tail Light |
| Turn Signal Indicators |
| Anti-theft Alarm with Motion Detection |

## 2.2 Out of Scope Objectives

These objectives are those which are outside of our scope due to limitations in regards to time, finances, development, and familiarity. These may also be referred to as our stretch goals, to be achieved after we have completed the In-Scope Objectives.

Table 2.2.1: Out Of Scope Software Objectives

| Software |
|---|
| GPS Location for tracking |
| Speedometer |
| Odometer |
| Collision Monitoring and Emergency Contacting |

Table 2.2.2: Out Of Scope Hardware Objectives

| Hardware |
|---|
| Solar powered recharging |
| Bicycle powered recharging (Power from wheels spinning) |
| Lock and key for alarm or electronic remote |
| Collision Detection |

## 2.3 Functional Requirements

These are the actual requirements which we will fulfill in pursuit of our objectives. These contain more specific information in relation to current standards or most commonly used versions. Some are simply self-imposed requirements created in order to quantify objectives.

Table 2.3.1: Functional Software Requirements

| Software |
|---|
| Support iOS and Android (iPhone 8+, Android API 28+) |
| Maximum size is 1GB |
| Maximum RAM usage of 512MB |

Table 2.3.2: Functional Hardware Requirements

| Hardware |
| --- |
| USB recharging (Type B or C connector) |
| Headlight output of at least 400 - 600 lumens |
| Tail and signal lights of at least 50 lumens |
| 2000mAh Rechargeable Battery |
| Operational for 5 - 10 hours, Recharges within 2 - 4 hours |
| IPX6 Waterproof |
| Alarm at least 100dB |
| MCU Architecture 16-bit or greater |

Further explanation and exploration of requirements will be conducted in later sections in order to more deeply expand on the content. It is there that design choices will be discussed and the implementation will be decided.

## 2.4 House of Quality

As the project design continues, positives and negative qualities related to the engineering qualities of the project will be compared. House of quality helps distinguish the requirements through a simple diagram. Each requirement is compared to another and ranked in priority.

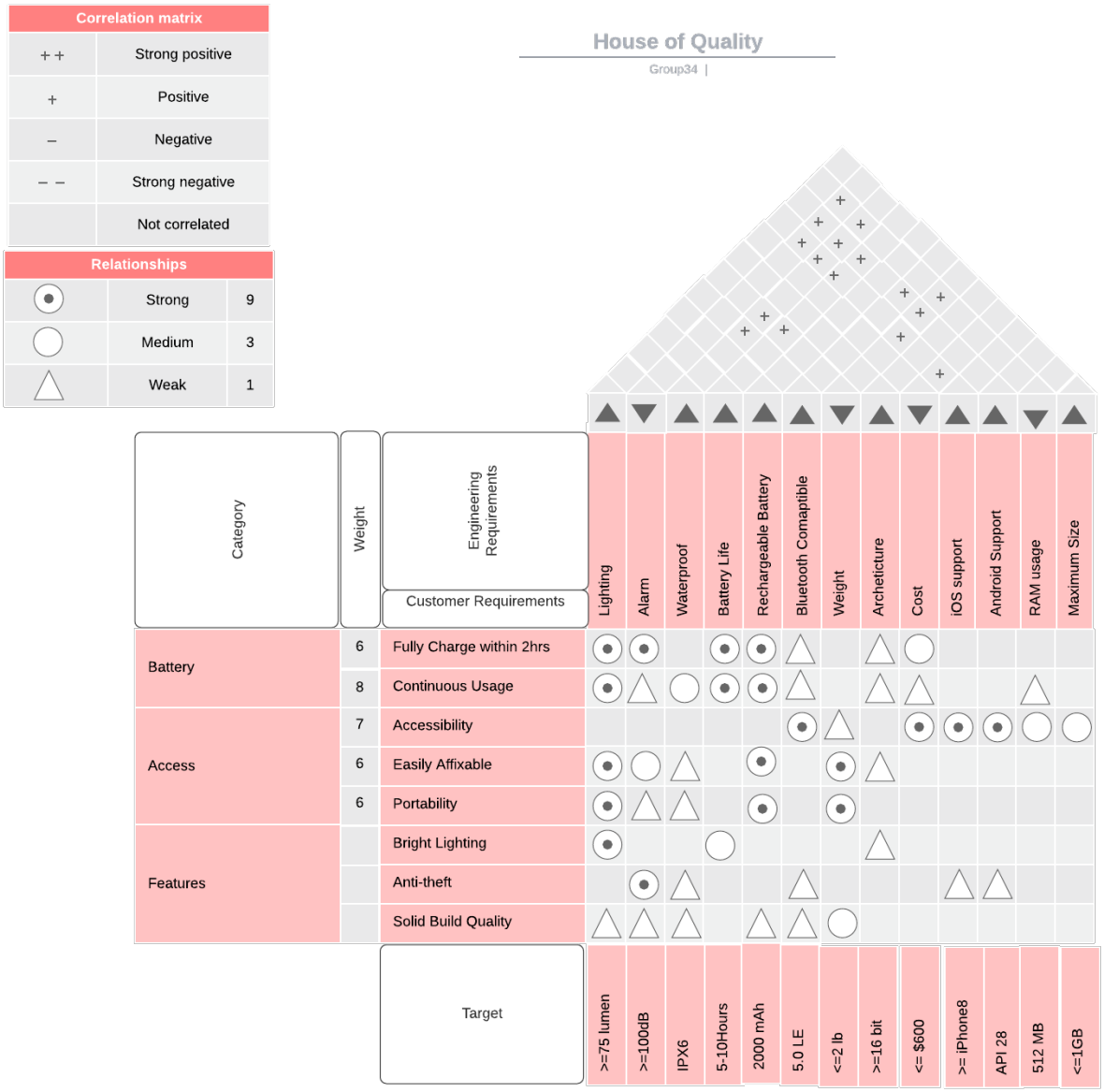



Figure 2.4.1: House of quality diagram

# 3 Research and Part Selection Technology Comparison

In this section, the assessment in all technology parts that will be utilized for this project will be further analyzed. This will substantially distinguish the various technology parts to help make decisions as to what will be used for the project.

## 3.1 Microcontroller

A microcontroller (MCU) is a powerful, yet small, versatile, and cheap device that is involved in almost every modern electronic product. Essentially at a basic level, a microcontroller is an integrated circuit device that controls certain aspects of a system. A microcontroller at its fundamental core includes a processor core, Random Access Memory (RAM), Read-Only Memory (ROM), Inputs/Output ports (I/O). So, in essence, a microcontroller is a simplified computer. Its main role is to do a single job and do it repeatedly. They're widely known for embedded applications, which can vary from anything like vehicle control systems to home appliances such as a dishwasher. The project will utilize a microcontroller to communicate with systems such as turn signals and the security protocol designed for anti-thefts. It will make decisions based on inputs and distribute voltage to enable certain outputs.

Regarding making a decision as to what type of microcontroller the project will use, there are multiple factors that need to be discussed. The core types of a microcontroller, of which there are three widely known types: 8-bit microcontrollers, 16-bit microcontrollers, and 32-bit microcontrollers, which will determine the power consumption, the processing speed, as well as the price. Another important factor is Bluetooth compatibility. Based on the choice of the type of microcontroller there are many family names like the popular ones which include: ARM, Microchip Technology PIC, Microchip Technology Atmel AVR, and Texas Instruments MSP430. Going into detail about each of these factors will help eliminate the microcontroller options and knock them down to the best candidates.

It's also important to note that for this project the team is considering the idea to split the system into utilizing two microcontrollers rather than having just one. This can help ease the load of the microcontroller utilization. The team could go with the route of having a microcontroller that will control the lighting system while the other microcontroller will consist of having a Bluetooth module, an alarm, and possibly a GPS receiver to communicate with a phone app.

Working on the microcontroller is expected to be one of the most time consuming parts of this project as having to get it work effectively would be a long task. The one that is chosen does not have to support parallel code execution but it would be nice if it had out of box support for concurrent code execution to keep response times minimal.

### 3.1.1 8-bit, 16-bit, and 32-bit

The key distinction between 8-bit, 16-bit, and 32-bit microcontroller types is the bus width. This specification limits the speed of calculations due to the smaller bit microcontroller option requiring more instructions to perform higher bit calculations. The slowest to fastest microcontroller type in order is 8-bit, 16-bit, and 32-bit. An 8-bit can typically run at 8 MHz while a 32-bit can achieve up to hundreds of MHz. Speed is crucial particularly in situations that involve mechanical relays and heavy data processing applications, noted in [3].

In relation to the project, a sweet spot between all choices is the 16-bit microcontroller, which provides the low power close to its predecessor, the 8-bit, while also maintaining performance regarding speed and memory like the 32-bit [4]. However, the biggest disadvantage of a purely 16-bit microcontroller is the fact that it does not offer Bluetooth compatibility. A 32-bit microcontroller is the only option left because it's the only type that offers Bluetooth wireless connectivity. Each type has its advantages and disadvantages, however, the project overall requires a microcontroller that has many low power options to help conserve power for the cyclist on long trips.

This analysis helped the team conclude that the team could use a 16-bit microcontroller as the lighting controller and a 32-bit microcontroller to communicate with the mobile app. Either that or the team will move forward with only using one microcontroller, specifically a 32-bit, to handle the entire system.

### 3.1.2 Microchip Technology PIC Microcontrollers

Up to now, the microcontroller needs to be 16-bit or 32-bit, low-powered, and cheap. Starting off with the PIC microcontroller family, particularly the PIC24F, it offers low power and low cost while also providing up to 1 MB memory and 16 MIPS operation at 32 MHz It's a cost-effective option to 32-bit microcontrollers, and its multiple power management options can allow the project electronic system to last longer, which is crucial for cyclists on long trips.

The multiple power reduction options include VBAT, deep sleep, sleep, and idle modes, and doze mode. VBAT is an option that involves running low power consumption upon the usage of a backup battery. While Deep sleep is basically a near-total power shutdown ability and wakes up upon external triggers. Sleep and idle, as the name suggests, shut down certain peripherals to reduce power consumption while maintaining fast wake-up time. Lastly, the doze mode is an option that lets the CPU run at a lower clock speed. The current consumption is incredibly low for deep sleep typically at 60 nA. In low power, the run mode is 160 µA / MHz [5]. There are a lot of low-power options to work with.

The PIC24F 16-bit microcontroller compared to its family members is summed in Table 3.1.2.1. The three types of program memory size within the PIC24F is 64KB, 128KB, and 256 KB. The RAM size can either be 8K x 8 or 16K x 8. The options for the number of I/O include 52, 84, and 101. All PIC24F family members have FLASH program memory type, 16bit core size, PIC core processor, and the voltage supply range from 2V to 3.6V.

Table 3.1.2.1: PIC24F Differences Based on Model

| Name | I/O Ports | Program Memory Size | RAM Size | Price |
|---|---|---|---|---|
| PIC24FJ256GB406-I/PT-ND | 52 | 256KB | 16K x 8 | $6.730 |
| PIC24FJ64GB406-I/PT-ND | 52 | 64KB | 8K x 8 | $6.160 |
| PIC24FJ128GB406-I/PT-ND | 52 | 128KB | 16K x 8 | $5.336 |
| PIC24FJ256GB412-I/BG-ND | 101 | 256KB | 16K x 8 | $7.890 |

It's also important to note that Microchip offers a free Integrated Development Environment (IDE) called MPLAB. The IDE allows the use of the C programming language, which all members of the team are familiar with. This will help because none of the members must spend time learning an unfamiliar language to program the microcontroller. The MPLAB offers a lot of analysis tools to minimize development time as well as design resources like software libraries and datasheets.

Overall, this choice is solid if the team decides to go with the two microcontrollers route. Its use will serve as the lighting controller.

### 3.1.3 ESP32

The ESP32 is a general-purpose 32-bit microcontroller that has both WiFi and Bluetooth wireless connectivity. It is relatively cheap and offers ultra-low power consumption with its various power modes and dynamic power scaling. Its microprocessor clock rate can achieve up to 240 MHz according to the datasheet [6].

The ESP32 has a hybrid WiFi and Bluetooth chip that can operate through SPI / SDIO / or I2C / UART interfaces [7], which all can be seen included in Figure 3.1.3.1. It is Bluetooth Low Energy (BLE) compatible, which is great considering our ultimate goal is to cut as much power consumption as possible to preserve battery life. The base OS is FreeRTOS, which has many benefits such as being free, Internet of Things (IoT) libraries, code compatible with various architectures like ARM-based microcontrollers, and of course the fact that it is open-source [8] [9]. Another bonus for the ESP32 is that Arduino IDE and NodeMCU can be used, which although is not the best environment, serves as an advantage considering its learning curve is great for beginners.

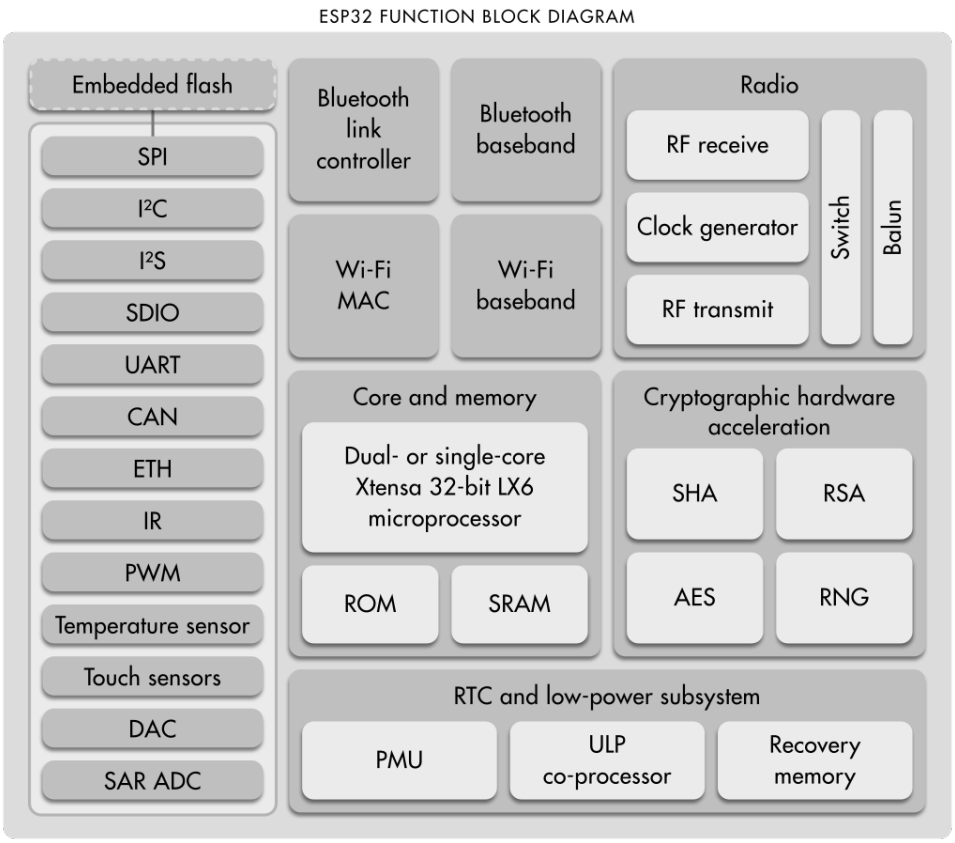



Figure 3.1.3.1: ESP32 Function Block Diagram [6]

To summarize the family members that are part of the ESP32 Table 3.1.3.1 is created to help distinguish the differences. All are microcontroller System on Chips (SoCs), have Wifi IEEE 802.11 b/g/n; 2.4 GHz however vary in speed, and include Bluetooth but some support different versions.

Table 3.1.3.1: ESP32 Differences Based on Model

| Name | Bluetooth | GPIO | FLASH (MB) | SRAM (KB) | ROM (KB) | Freq. (MHz) |
|---|---|---|---|---|---|---|
| ESP32-C3 | Bluetooth LE v5.0 | 22 | 0 | 400 | 384 | 160 |
| ESP32-C3FN4 | Bluetooth LE v5.0 | 22 | 4 | 400 | 384 | 160 |
| ESP32-C3FH4 | Bluetooth LE v5.0 | 22 | 4 | 400 | 384 | 160 |
| ESP32-D0WD-V3 | BR/EDR + Bluetooth LE v4.2 | 34 | 0 | 520 | 448 | 240 |

The ESP32 is a strong candidate that offers a lot of what the team is looking for especially considering it offers WiFi and Bluetooth Low Energy technology. It also is relatively cheap and well supported.

### 3.1.4 CC2640

Another candidate is the CC2640 SimpleLink, which holds a 32-bit ARM processor that can achieve up to 48 MHz clock speed. It offers ultralow-power with its low power mode that makes MCU current very low benefiting battery life longevity. All important features that are part of the CC2640 microcontroller are shown in Figure 3.1.4.1, where general peripherals, modules, and memory information are shown. It also holds a Bluetooth Low Energy controller, which is beneficial for the project system.

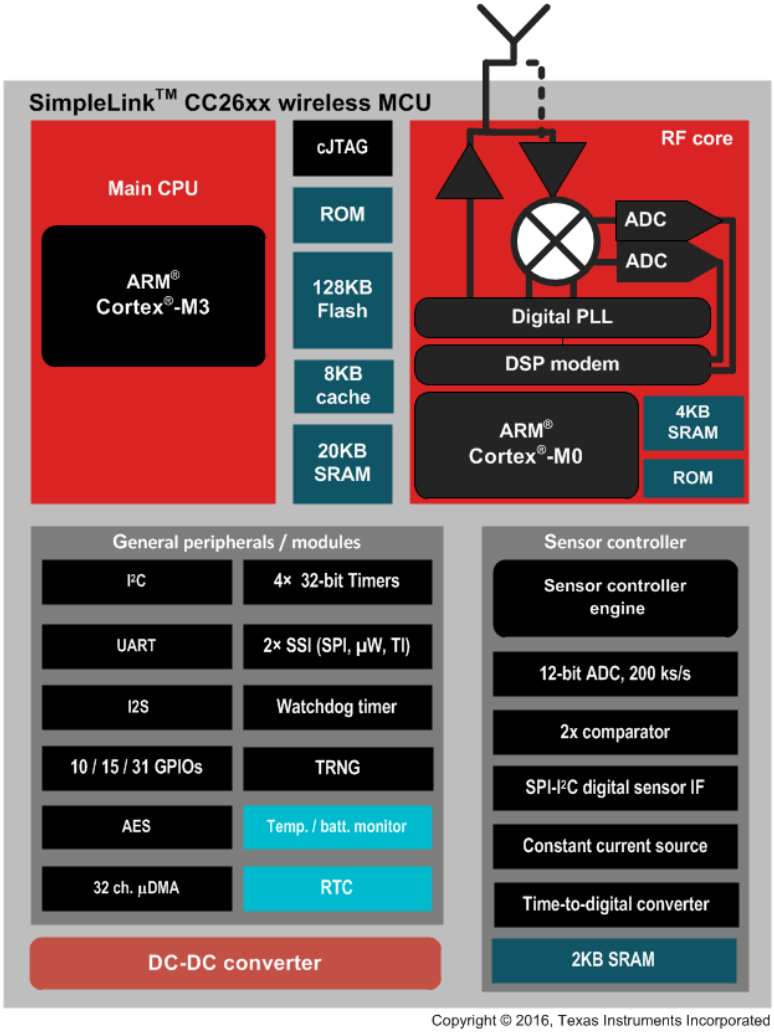



Figure 3.1.4.1: Functional block diagram for the CC2640

The CC2640 features:

- Bluetooth Low Energy 5.1

- 20 KB RAM
- 10,15,31 GPIO
- Peripherals contain:
  - 1 UART
  - 12 bit ADC 8-channel
  - 2 SPI
  - 2 Comparators
  - 4 timers
  - 8-bit DAC
  - I2C
  - I2S
  - Sensor Controller

The CC2640 Ultralow-Power Sensor Controller can run autonomously while the rest of the system is in sleep mode [9]. This is incredibly useful in applications that require collecting analog and digital data for an extensive amount of time. The battery life can be extended by utilizing this option. In regards to the low power mode, normal operation ranges from 1.8 to 3.8 V. Standby mode consumes 1 µA while RTC is running and RAM/CPU retention. Lastly, the shutdown current consumes 100 nA.

In regards to the simplicity of programming the CC2640, Code Composer Studio will be used since the device is a Texas Instruments product. The advantage of this is that all members are already experienced with programming with the Code Composer Studio IDE.

### 3.1.5 Microcontroller Comparison Breakdown

Table 3.1.5.1: Microcontroller Comparison Breakdown

| Microcontroller | PIC24F | ESP32 | CC2640 |
|---|---|---|---|
| Type | 16-bit | 32-bit | 32-bit |
| Pin Count | 28-121 | 48 | 48 |
| CPU | PIC24F | Xtensa | ARM Cortex-M3 |
| RAM Memory | 8KB - 32KB | 320 KiB | 20KB |
| Bluetooth | N/A | 4.2 BR/EDR and BLE | BLE |
| Wifi | N/A | 802.11 b/g/n | N/A |

## 3.2 Anti-Theft Alarm

The Alarm is an important part of our system and serves the specific function of theft prevention. There are several considerations to be made including sound level, size, and connectivity. Our ideal speaker is loud enough to be heard from a distance, small enough to fit onto the final product, connects easily into our electrical circuitry, and is safe from tampering. There is also the option of purchasing a buzzer which performs the same purpose as the speaker but can only emit a loud alarm noise when powered.

The alarm is used as a deterrence for a would-be thief, similar to home and car alarms. The sound produced alerts the owner and draws attention from bystanders, which makes the carrying out of the action much more difficult for a thief. The alarm would have to be designed to emit a sound at a pitch similar to a common alarm so that there is a more ubiquitous idea that this is an anti-theft signal.

### 3.2.1 Sound Level

Searching for Bicycle alarms online reveals that all are rated for levels above 100dB, with quite a few exceeding 110dB. So our product should ideally match that. By comparison, the sound level at which a conversation is held is approximately 60dB, whereas shouted conversation is at about 95dB. Decibel levels above 85 dB are considered harmful to the ear if under extended exposure, however, this is not a concern (see section 4) as the alarm is meant to be used in a brief period. This can be related to a car alarm; a car alarm is loud and irritating, but the sound is meant to alert the owner and deter a potential thief rather than do any damage to the ear.

### 3.2.2 Size

All of the speakers we considered are very small, being less than a few inches in most dimensions. These should produce enough sound to be heard several feet away and would not draw too much power.

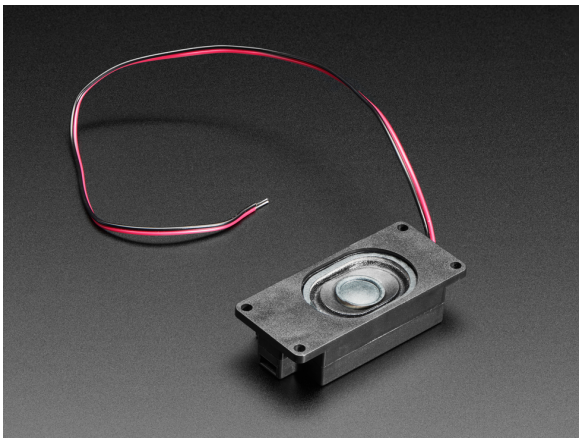


Figure 3.2.2.1: Mono Enclosed Speaker [10]

Figure 3.2.2.2: Speaker 0.5W [11]

The speaker on the left is a mono-enclosed speaker, which draws 2 Watts of power and 6 ohms of resistance. The rated voltage for this size averages about 3.5 V. The speaker on the left draws 0.5 Watts of power and 8 ohms of resistance. Thus, the two speakers are off by a measurable degree, but both could be utilized in the design with no noticeable issues; the power consumption will just have to be properly accounted for.

For both speakers, power consumption is not a significant constraint on design; most externals for microcontrollers are not designed to suck up power, and can usually support a low-power mode (which will be utilized in the design). Also for both speakers, weight is not a concern or a constriction. The weight of each speaker is less than 50 grams, making the addition to the microcontroller setup nearly unaffected; as with the power consumption it just needs to be properly accounted for in the final design report.

Typically, microcontroller speakers are designed for simple signals such as beep codes. Attaching higher quality speakers allows for more advanced signals (assuming the method of communication is supported). However, for this design, a high-quality speaker is not required. The alarm system that will be utilized does not need to be high-quality (e.g., crisp and clear) sounding; the purpose of the alarm is to produce a loud ringing for deterrence and protection (i.e., decibels produced are the only potential factor of quality).

### 3.2.3 Connectivity

There are a few ways to connect a speaker to our system. USB, direct wiring, and auxiliary are the main options. These options are the most common and are likely to be found on our selected microcontroller. The speaker in Figure 3.2.2.1 is 3W and requires 4Ω impedance while the speaker in Figure 3.2.2.2 is 0.5W and requires 8Ω impedance.

### 3.2.4 Anti-Tampering Environmental Protection

One major concern is that a potential thief may easily destroy a speaker, so it must be protected not only from the environment but also from tampering. This may be achieved by placing the speaker within the main product casing and spraying the speaker with waterproofing.

## 3.3 Battery

The battery is the lifeforce behind the entire product and must thus be considered very carefully. Our requirements state that we must have something at least 2000mAh in capacity which recharges within two hours. Further considerations include methods of charging, like what port and how it connects, along with outputs including voltage and current. In order to create a rechargeable power system, we will utilize lithium ion (li-ion) batteries which are widely available and very commonly used for a variety of applications. Finding a battery or batteries that give us the required mAh is not difficult, a simple search reveals that one may even easily acquire a battery rated for 10000mAh. With this in mind, our main concerns when considering which rechargeable battery to use are connections and output levels. [12] [13]

While considering which battery to use we first had to figure out which type of battery to use. There are many types of rechargeable batteries which each have their advantages and disadvantages. The most popular types are Lithium-ion (Li-ion), Lead-Acid (Car Batteries), Lithium Polymer (Li-po), Nickel-Cadmium (Ni-CAD), and Nickel-metal-hydride (Ni-MH).

Due to the high self-discharge rate, Lead-acid batteries are not a good choice. Though they do provide a lot of energy and may survive hundreds of monthly charge cycles, they are overkill and are not the best suited for this application. We expect our product to at most be charged once per day, this equates to roughly thirty charge cycles per month which is far under what a Lead-acid would be able to easily provide but it is more than likely that a battery of this type would not need to be charged more than once a week or perhaps even once a month due to the sheer amount of energy they store. A downside to this is that, in the case of leaving a car battery sitting for a few weeks, the battery will die and it must be jumped which makes the product a lot less viable for the everyday user. Further, while Lead-acid batteries do provide a solid voltage generally of 6V or 12V, the necessary charging voltage makes it a more difficult choice to build for and decreases product simplicity. Beyond the simpler issues related to weight and dimensions of these batteries, this is simply not a good choice for our intended application.

In the cases of Lithium Ion and Lithium Polymer, we see two very commonly used types especially in electric vehicles and portable electronics such as smartphones and laptops. These batteries provide a high energy density which makes them compactable while still maintaining a significant energy storage. Li-po specifically is mouldable which allows it to fit into more size-constrained applications but is also fragile. Li-po batteries also weigh less than Li-ion but sacrifice energy density. These batteries are very low maintenance and charge very quickly as well. The main downside for both of these is that they are much more expensive and will negatively impact our market reach and accessibility. Further, improperly handling or charging/discharging may cause one of these batteries to burst into flames. This means that a protection circuit must be put into place to protect the battery from irregular voltage, current, and temperature. These batteries decrease our product safety and thus require us to not only spend more on the battery itself but also on systems and features which will protect the product against weather and isolate vital components from battery failure. These batteries do boast an extensive amount of charge cycles which is great for our product lifecycle, Li-ion and Li-po batteries would very likely have a smaller capacity when compared to the lead-acid which means we could expect this battery to go through a maximum of one charge cycle per day with an expected usage of one to two charge cycles per

week for a rider who uses their bike for approximately one to two hours per day. Further, the self-discharge rate is much lower and thus makes this a better option for a light-use application such as ours. Actual charging of these types of batteries would be made easier with the use of USB but implementation is more difficult due to the aforementioned protection circuits. These are also generally a lower voltage than the lead-acid, less than 5V, so they would further require an amplifier or step-up circuit. This is a solid choice for our product but it leaves more to be desired.

Lastly, we have Nickel-Cadmium and Nickel-Metal-Hydride, these two types are very similar and single cells of either are rated at 1.2V. This means we would have to connect multiple in series and possibly create parallel banks in order to establish our required voltage and current. Ni-CAD batteries date back to 1899 in Sweden, so they have a long history and have been widely used for over a century. It has been mostly replaced by Li-ion and Ni-MH since their release in the 1990s but remains a viable option. Ni-MH was developed to reduce toxic substances used in Ni-CAD along with increasing the capacity, giving a thirty to forty percent boost. Both types suffer from what is called the memory effect which results in the battery delivering less energy than its rated full capacity after just a few partial discharge cycles. This may be overcome by fully discharging each cell and then fully charging it. Ni-MH batteries suffer this effect to a much lesser degree and as such were marketed as being free from the memory effect altogether. In terms of self-discharge, Ni-MH takes the cake as it performs at the level of Li-ion if not higher, meaning it more sufficiently holds its charge. Ni-CAD has a rate between lead-acid and Li-ion which makes it acceptable but not ideal. Both of these types do however have a very high cycle durability which would greatly increase our product's longevity while also being easy to replace as they come in standard sizes such as AA and AAA. The Ni-MH type does appear to be more sensitive to temperatures in relation to self-discharge so this could be very problematic when a rider is out in the heat or directly under the sunlight, which many riders tend to be. However, due to our product being intended for nighttime use, this may be an acceptable downside. These types are generally much safer than their Lithium and lead-acid counterparts as they are more resistant to leakage and explosion. These batteries are more tolerant to overcharge and if maintained properly may have a greater longevity. They do provide greater shelf-life which is great for particular applications such as fire alarms, or in our case, an anti-theft alarm. The biggest downside to these types of batteries is that there is a great difficulty in charging them. While they do not require a protection circuit like Li-ion batteries do, the best method for charging them is a slow charge. This means that the recommended charge time for these types of batteries is going to be roughly six to seven times larger than our requirement and would thus make these types completely incompatible with our design. Fast charging results in a heavily reduced charge time well within our requirement specification but at the cost of some of the battery's capacity. The other issue with regards to charging is that, in the case of slow charging, the batteries must be manually disconnected, so a timer or alert would be necessary to ensure disconnection by the user. In the case of fast charging, advanced circuitry would have to be implemented in order to prevent the batteries from being overcharged and damaged.

There are three main options for charging these batteries. Trickle charging, which is a fixed and low current less than $1/10$ of the capacity, allows for the greatest reduction in risk in regards to overcharging but requires an acceptable amount of recharge time for the product. The fast charging methods include delta V, which monitors the change in voltage of the battery over time.

This method realizes that a fully charged battery causes the voltage across the terminals to dip slightly, once detected the charger disconnects, for Ni-MH the drop is much smaller and can be impossible to detect in a practical and reliable manner. The other fast charging method is delta T which is similar to delta V but instead realizes that when a cell is fully charged, the energy being put through it will no longer convert into chemical energy but will instead convert into heat, raising the temperature. Panasonic and Duracell suggest a maximum delta T cutoff of one degree Celsius per minute. Both of these require more advanced electronic circuitry or the purchase of a charge regulator. Being able to purchase a charge regulator would greatly ease our development, while allowing for more efficient fast charging. Overall, both of these types are very viable for our product and seem to fit between Lithium and Lead-acid batteries in a lot of respects, making them a couple of the top contenders.

### 3.3.1 Output

A lot of rechargeable batteries have ports for USB charging or they have loose wires which are meant to be connected directly to the circuit, others simply have terminals with which to connect your own circuitry. Depending on the microcontroller that we use, we will have to either do a mixed combination of the aforementioned types or we will have to route the circuitry through a couple of voltage dividers in order to achieve those levels necessary for not only the microcontroller but also the light system, this is depicted in Figure 3.3.1.1.

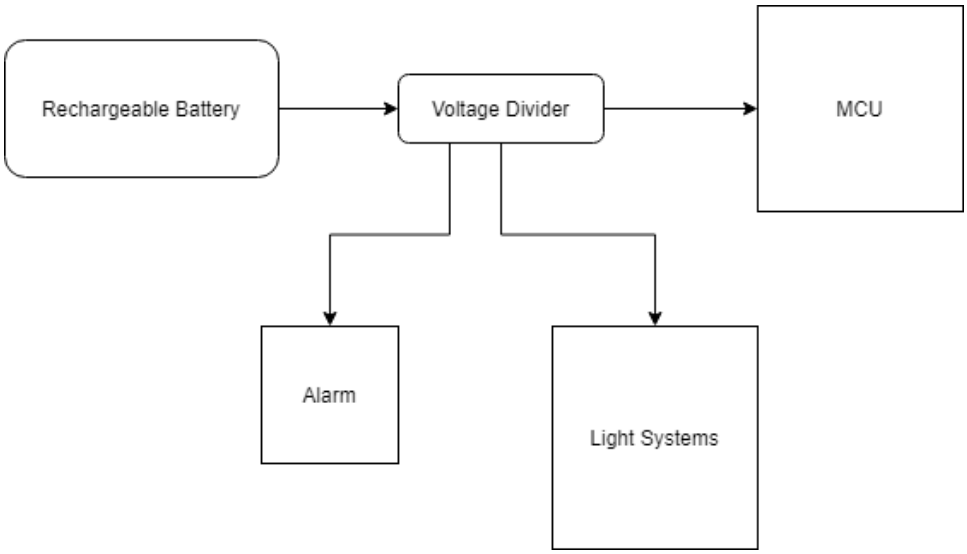



Figure 3.3.1.1: Diagram for High Voltage Rechargeable Battery

Some options we have are to use a higher voltage along with the dividers previously mentioned or a lower voltage with a higher current output to compensate for the necessary voltage amplification which is depicted in Figure 3.3.1.2.

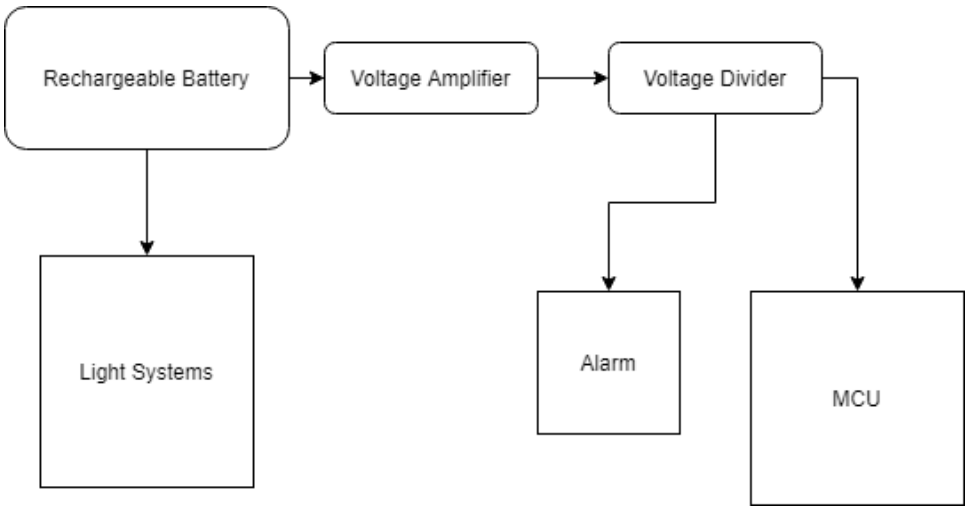



Figure 3.3.1.2: Diagram for Low Voltage Rechargeable Battery

It is possible that by utilizing Nickel batteries that we may achieve a voltage approximate to that which is necessary to run our system. This may be achieved by placing multiple cells in series. In order to achieve our desired current, we may increase parallel cell series.

Most microcontrollers require a 5V input and have regulators for other necessary voltages such as 3.3V, this is the case with the ESP32 which has a 5V input. The CC2640 however has an input range of 1.8V to 3.8V. Lights can range from 3V to 48V, with the higher end being common in Electric Bicycles. 12V appears to be the standard for regular headlights, perhaps owing to the fact that most batteries (the type that would be used in a car) output 12V. A higher output will require more power, so we expect that the lights will consume a majority of the battery's energy and will be the main contributor to lower battery life. MCUs are developed such that they have a variety of low-power modes and most are created with their applications in mind such that their overall power consumption is already reduced. Our Alarm will be inactive the vast majority of the time so it will not affect the battery life too much.

Those batteries which include an output of USB are Type A female, this means that the MCU must also have a USB connection whether that be micro-usb type B or type C and must thus be one of the common types of USB cabling. Some batteries also include an output of USB Type C. There are also options to use a battery holder which may use a DC connector or direct wiring, this option allows the use of multiple smaller batteries that may conform better to our design. A battery holder would allow the user to more easily replace faulty batteries while further providing redundancy in the case of the failure. A battery holder is the more viable option for this project as it allows us to wire the batteries directly up to the lighting system as opposed to some kind of USB to DC connection.

### 3.3.2 Input

Depending on which type of battery we use, our input will vary. Our most likely inputs are Micro-USB Type B and DC. As discussed in section 3.3, if we select either of the Lithium type batteries we will need to implement a protection circuit against irregular Voltage, Current, and

Temperature. If we select either of the Nickel type batteries we will need a protection circuit to prevent overcharging while fast charging.

Charging is measured as a portion of the battery's overall charge, represented as C. For a 2000mAh battery that is Nickel-based, the recommended current is C/10 or 200mA for a slow charge. For a fast charge, 1C may be applied or 2A. This would allow the fast charge to meet the requirement for a less than two hour charge time. For Lithium-based batteries, the charge rate may be between 0.5C and 1C which, on the higher end of the range, should allow us to meet our requirement as well.

### 3.3.3 Backup Battery

Our backup battery system is necessary to keep the anti-theft alarm system operational in case of main battery failure or complete discharge. It is with this in mind that we must select a battery type that may output the correct voltage with a voltage divider or step-up circuit as necessary. This battery type must have a long shelf life and thus a self-discharge rate that is extremely low. This alarm will function similarly battery-wise as a fire-alarm in that until the alarm is triggered, the power draw on the battery is kept to a minimum and is as such practically zero. The main contenders for this battery are the Lithium type and Alkaline type.

Alkaline batteries generally have a long shelf-life at several years if not longer, this is perfect as the backup battery system should require very infrequent replacement. Output is advertised at 1.5V or 9V which may require a voltage divider or step-up circuit to fully power the alarm. These batteries are very widely available and come in many sizes including D, C, AA, AAA, AAAA, and 9V, see Table 3.3.3.1 [14]. Size will depend on the speaker requirements. This is a very ideal choice due to price, wide availability, and support.

Table 3.3.3.1: Battery type dimensions

| Battery Size | Diameter (mm) | Height | Weight |
|---|---|---|---|
| D | 32 | 61 | 140 |
| C | 25.5 | 50 | 72 |
| AAA | 10.5 | 45 | 13 |
| AAAA | 8.3 | 42.5 | 10 |

Lithium batteries have an even greater shelf-life than their Alkaline counterparts lasting up to twenty years and as such are often relied on in important electronics such as watches and medical devices. These batteries are not to be confused with Lithium Ion and Lithium Polymer which are rechargeable types. Lithium batteries do cost significantly more than Alkaline, often costing a minimum of three times more. They are very widely available, similar to Alkaline and come in the same standard sizes as mentioned above. Again, size would depend on the speaker requirements. The increase in cost alone is enough to make this a poor choice for our product.

A table found from Benzo Energy summarizes the differences between Lithium Ion and Lithium Polymer [15]. This table is shown in Figure 3.3.3.1.

| Basis | Lithium Ion | Lithium Polymer |
|---|---|---|
| Ageing | Loses actual charging capacity over time | Retains charging capacity better than Lithium =ion |
| Energy Density | High Energy density | Low as compared to lithium ion |
| Conversion Rate | The capacity to convert battery into actual power 85-95% | 75- 85% |
| Safety | More Volatile as compared to lithium polymer | More safety. Less chance of explosion |
| Cost | Cheaper | Slightly Expensive(+30%) |
| Weight | Heavier | Light Weight |
| Charging duration | Longer Charge | Comparatively Shorter |

Figure 3.3.3.1: Lithium Ion vs Lithium Polymer [15]

## 3.4 Lighting

Lighting will be crucial for the entire project system. At night, cyclists are incredibly vulnerable to danger due to visibility issues from those driving in cars. Thus, the lighting system will provide clear visibility for cyclists at night time to ultimately increase safety and protection from harm. To accomplish this, the lighting system will cover illumination in the following sectors: brake lights, turn signals, and headlights. The lighting system can be achieved in various ways and each lighting sector will require particular patterns and colors such as the brake light illuminating the color red to signal for stopping or braking. Selection of the type of technology for the lighting system will also be discussed to further analyze what the group project will need.

### 3.4.1 Lighting System Defined

The lighting system will consist of brake lights, turn signals, and headlights. These different locations will help illuminate the entirety of the bicycle to increase visibility on the road. Brake lights will output red automatically upon the bike braking or stopping. Turn signals will be indicated by either flashing arrow patterned lights or simple yellow flashing lights. The headlights will illuminate a bright white color strong enough to be seen blocks away while also consisting of different modes such as low and high brightness levels. It's important to note that all lights will automatically sense the level of brightness to adjust to based on outside conditions. Such conditions can be rainy, daytime, and nighttime. There will also be options to choose from on the controller on the handlebar of the bicycle so customers that wish to disable, turn on, or select brightness level can manually do so.

### 3.4.2 Types of LEDs

There are a variety of LED types and researching the types will help conclude what type will be needed for each part of the lighting system. Some LEDs have advantages over others such as being addressable, smaller, and higher power.

#### 3.4.2.1 SMD LEDs

Surface Mounted Devices (SMD) is a type of Light Emitting Diode (LED) that requires no wires. The SMDs can be soldered on circuit boards and make the lighting system aesthetically pleasing rather than requiring bundles of wires to be run through. There are a variety of SMD LED options. Some of which have low power consumption, moisture sensitivity, and multicolor [16].

The SMD LED types vary in shapes and sizes. Common types are 2835, 3014, and 5050. The SMD LED type name represents the dimensions of the chip in mm, such as the 2835 SMD LED which translates to 2.8mm x 3.5mm. Depending on the type of SMD LED the wattage drain, as well as the amount of produced light, can vary. For example, again using a 2835 SMD LED, produces 20% more light but draws less wattage when compared to a 5050 SMD Specifically, a 2835 MSD pulls 0.2W while a 5050 SMD consumes 0.24W [17]. 2835 SMDs also take up smaller space meaning more can fit into an allocated space, which in return will produce more lumen per meter. Some comparisons between the type of SMD LEDs are shown in Table 3.4.2.1.1.

Table 3.4.2.1.1: SMD LED Chips Characteristics [18]

| SMD LED | Dimensions (mm x mm) | Power (Watt) | Flux (Lumen) |
|---|---|---|---|
| 2835 | 2.8 x 3.5 | 0.2 | 22 |
| 3528 | 3.5 x 2.8 | .08 | 4 |
| 5050 | 5.0 x 5.0 | 0.24 | 12 |
| 5060 | 5.0 x 6.0 | 0.2 | 18-26 |

#### 3.4.2.2 COB LEDs

Chip-on-Board (COB) LEDs are newer than SMD LEDs. COB LED technology is superior due to the amount of packing density that can be held in terms of the LED array. In simple terms, the higher the packing density the more LEDs that can fit within an array. This greatly increases the lumen density while reducing energy consumption. To further demonstrate this, in Figure 3.4.2.2.1 a comparison is done between different LED technologies.

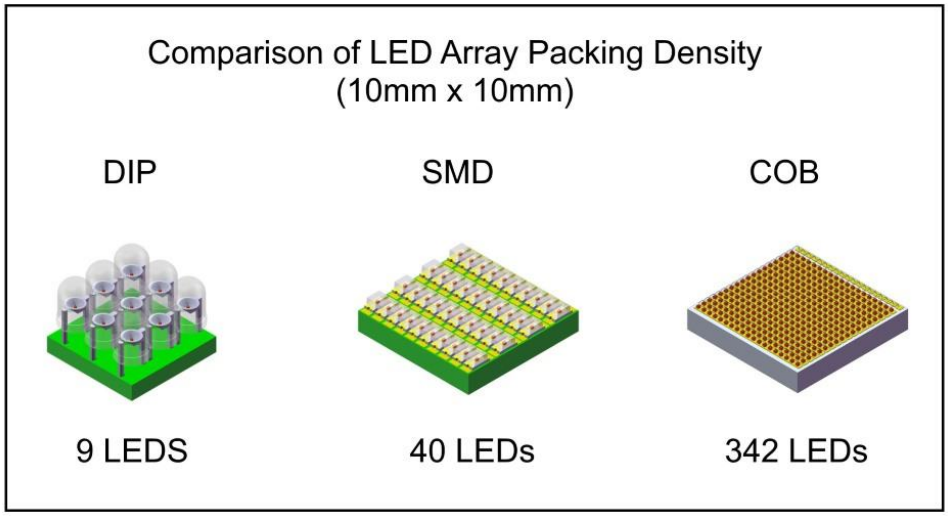



Figure 3.4.2.2.1: Comparison of LED Array Packing Density Among Different LED Technologies [19]

COB LEDs may be the best choice among the LED technologies simply due to brightness and low power consumption however, its drawback is the cost compared to SMD LEDs.

### 3.4.3 Brake Lights & Turn Signals

As mentioned in section 3.4.1, the brake lights will output a red color upon braking or stopping while turn signals will either illuminate blinking yellow lights or yellow blinking arrows. The brake lights are chosen to output a red color because most car drivers interpret stopping and braking with red lights. The color red is also associated with the sign of danger [20]. Red lights also can be seen further away compared to most colors because the frequency is the lowest among the visible light spectrum [21]. The brake lights and turn signals need to output at least 50 lumens and at least 180 degrees for safe illumination. Producing 50 lumens is enough to be visible at night time and vision hindered conditions. The group may want to increase this metric to increase competitiveness with other options since higher visibility will increase the likelihood of better safety from car drivers. The wider the angle of the light the better because this will increase the cyclists surrounding.

With that in mind, the best choice among the two LED technologies discussed in the previous sections would be COB LEDs because the brightness and low energy consumption are ideal. In regards to the design, it needs to be small and compact but big enough to produce the lighting specifications mentioned before. The team is considering the signal lights screen dimensions to be 2 inches by 2 inches by 0.5 inches (Length x Width x Height).

### 3.4.4 Headlights

The lighting system will also cover the headlights in which the color will be white and have high lumens so the bicycle rider can see sufficient distance ahead. The rider may use the product for

not only lit urban areas but for unlit roads. Considering this as a possibility leaves the concern for the project headlights to be 400+ lumens for maximum visibility in dark unlit roads.

## 3.5 Switches

As mentioned in (3.4.3), the rear of the bicycle will contain LED screens used to indicate to traffic (i.e., cars and pedestrians behind the cyclist) that the cyclist is intending to turn left or right, and to indicate the cyclist is trying to come to a stop (exactly how signals work in motor vehicles). The turn signals will be controlled by two switches (buttons) located on the top of the handlebars on the left and right side. Where motor normally vehicles have a single 'bar' used to activate the turn signals, having two reduces the mistake of indicating the wrong direction to traffic.

The buttons do not connect directly to the turn signals, but are connected through to the microcontroller that sits in between the 'input' (buttons) and 'output' (turn signals); pushing the buttons would tell the microcontroller the appropriate action to take. The buttons are connected to the microcontroller by two wires that would connect to one of the input ports of the designated part of the microcontroller; while most higher-level (e.g16-bit and up) microcontrollers have buttons attached (or allow for buttons to connect easily), the microcontroller buttons are usually very small (less than 10 mm in any dimension). This can be easily overcome by wiring any commercially available electric button to the ports either on the microcontroller or the components of the original button.

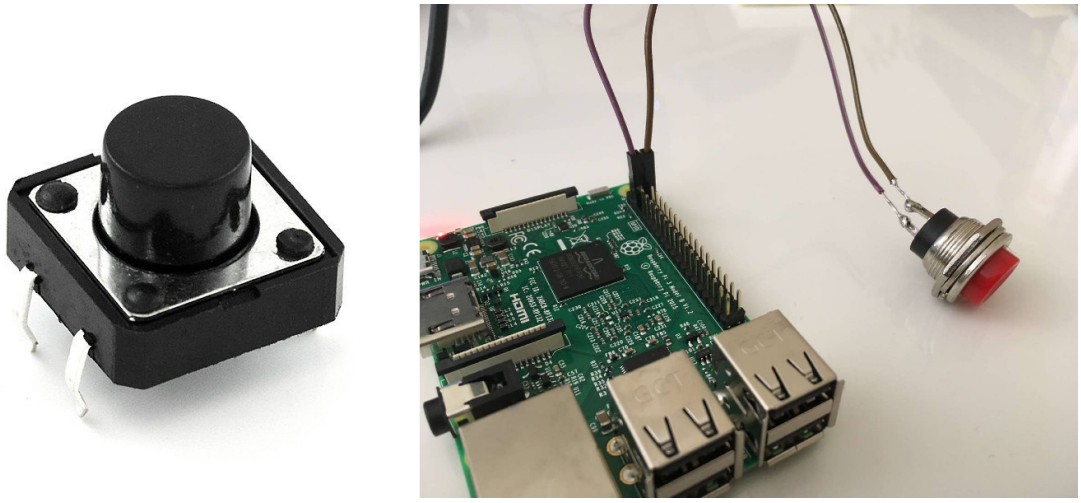


Figure 3.5.1: (left) a common microcontroller button (right) an example of how an electronic button of any size can be attached to a microcontroller

Using electronic buttons is a simple method of using mechanical force to complete a circuit; the button 'head' is held typically by a small spring or other mechanical restraint that completes an electric circuit when pushed. Common microcontrollers are able to program the method of button interaction to either "pull up" or "pull down" mode; which either completes the circuit or shorts it when the button is pushed.

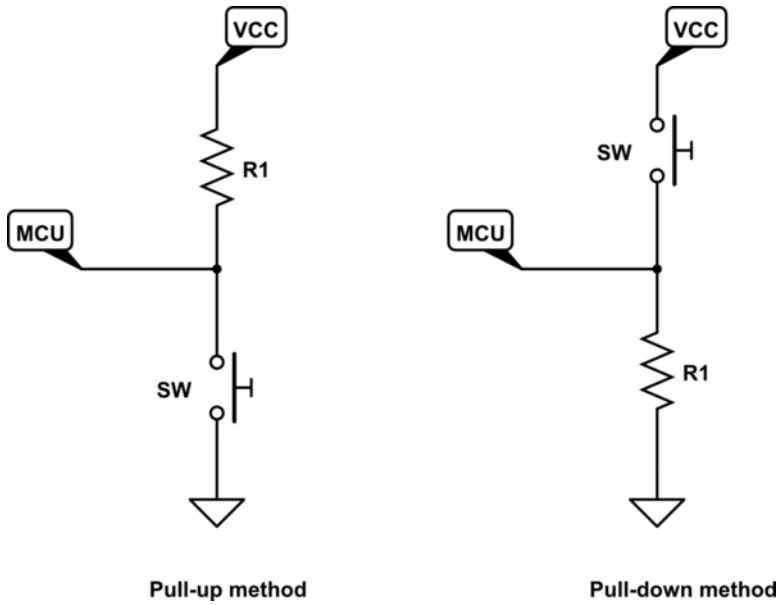



Figure 3.5.2: Basic pull-up and pull-down diagrams for microcontroller buttons

The buttons used in this design will be a pull-up method, meaning that pushing the button activates the signal to the microcontroller.

## 3.6 Communication Network

While there are 7 layers to the OSI model, only the physical layer and the application layer will be discussed as the features of the other layers are determined by either or both and to prevent repetition. The link layer is closely tied to the physical layer, which is unsurprising due to their proximity whereas the choice of the transport layer protocol is tied to the choice of the application layer protocol. For example, if the HTTP/3.0 (currently a draft) application layer protocol is used, then the QUIC protocol, which is a transport layer protocol is used, which is in contrast to the HTTP/2.0 application layer protocol, which uses the transport layer protocol TCP. Likewise, if a Bluetooth chip is chosen for the radio transmission through the physical layer, then the Bluetooth Link Layer will also be used.

Table 3.6.1: 7 Layers of the OSI Model

| OSI Layers |
|---|
| Application Layer |
| Presentation Layer |
| Session Layer |
| Transport Layer |
| Network Layer |
| Link Layer |
| Physical Layer |

For this project, a wireless communication protocol is required as it is not feasible to have a wired connection for the mobile devices, due to the lack of dedicated wired connection ports, the lack of uniformity in the multipurpose connectors the majority of the devices use and convenience that wireless connections provide by not requiring the user to connect their device for each use.

### 3.6.1 Physical and Link Layer

The physical layer is the lowest layer of the seven layers in the OSI model for computer communications. This layer is used to classify the physical connection among the devices in the network. The data is transferred in this layer with pure bits. There is no level of abstraction on this layer unlike the layers that are higher than this one. The hardware connection, the radio wave frequency, the thickness of the wires, the number of wires, the shape of the connectors for wired connections, and the type of radio wave are all described by the physical layer.

The Link Layer is the layer above the physical layer and therefore closely tied with the physical layer. The components that implement the communication protocols typically implement both of them, hence this section discusses the two.

#### 3.6.1.1 Wi-Fi Protocols

Wi-Fi is a set of standard protocols based on the IEEE 802.11 specifications. The following section discusses the most commonly used specifications of Wi-Fi networks.

**3.6.1.1.1 IEEE 802.11 Standards** The IEEE 802.11 is a subset of the IEEE 802 family of standards that are used for LAN (local area networks), PAN, and MAN. For the use of this design, the LAN is used as PAN is too small of a network and MAN is much too large. This set of protocols specifies the Medium Access Control (MAC) protocol of the link layer and the physical layer for Wireless Local Area Networks (WLAN).

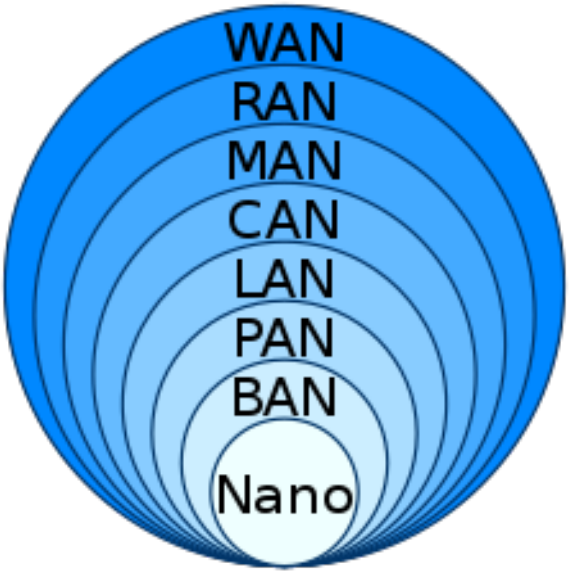



Figure 3.6.1.1.1: Scope of computer networks

The 802.11 is a standard for wireless LAN that will be integrated into the design for use through the smartphone app used to control actions of the microcontroller. Within the 802.11 protocol is a list of further sub-systems that are given letters to denote differences in their streams of data, bandwidth, and frequency band. The most accessible of these, and the one being used in this design, is the 802.11n protocol, which is able to operate in a frequency band of 2.4 to 5 GHz and operates either at 20 MHZ or 40 MHz depending on the data rate; 20 MHz operates between 7.2 and 72.2 mbps and 40 MHz operates between 15 and 150 Mbps [22].

#### 3.6.1.2 Bluetooth

Bluetooth is a wireless communication standard that is used to transfer data among devices over a short-range of distance. Bluetooth uses high frequency waves for communication and is widely used. The hardware and the software support for the protocol is also found on almost all mobile devices. Bluetooth being a family of protocols has most of its protocols use a server/client-like architecture.

Depending on the version of Bluetooth or the family, the ranges can be anywhere from 10 meters to 1969 meters. As with other radio waves, the longer range typically means reduction in speeds at which data can be transferred so picking the right parameters is crucial especially when choosing a dedicated Bluetooth module. This can be precisely calculated using the parameters used to define the physical layer like radio wave frequency, transmitter receiver gain, etc, on the website [23] that the Bluetooth Special interest Group hosts.

A paper published in the analysis of Bluetooth technology was used for its table that distinguishes the Bluetooth versions [24]. This is shown in Table. 3.6.1.

Table 3.6.1: Comparison of Bluetooth versions [24]

| Version | 4.0 | 4.1 | 4.2 | 5.0 & 5.1 |
|---|---|---|---|---|
| Multi-Roles | No | Yes | | |
| PDU Payload | Up to 31 bytes | | Up to 255 bytes | |
| LE Secure | No | | Yes | |
| IoT Support | Limited | | Medium | High |
| Advertising Channels | 3 Channels | | 3 Primary Ch.<br>37 Secondary Ch. | |
| Data Rate | 1 Mbps | | 2 Mbps | |
| Effective Range | 50 m (Line of Sight)<br>10m (Indoor) | | 200 m (Line of Sight)<br>40m (Indoor) | |
| Battery Life | Shorter | | Longer | |

### 3.6.2 Application Layer

Application Layer is the highest layer of the 7 layers of the OSI model. They provide the highest level of abstraction needed for communication among computers. Some of the application layer protocols include HTTP, MQTT, IMAP, TLS, SSH etc. These protocols can be implemented via code as long as the host environment provides support for the dependency layers below the application layer.

#### 3.6.2.1 Message Queuing Telemetry Transport

Message Queuing Telemetry Transport (or MQTT) is a widely-used, application layer protocol for embedded systems and Internet of Things (IoT). The protocol is suited for connections that have limited bandwidth and have low computational power [25].

Since the protocol is designed to be barebones, the protocol does not need to have a specific transport protocol and can be configured to work with any protocol that can have at least half-duplex communication, with proper error detection to avoid loss of data and in-order reception of data sent.

To keep the power footprint low, this protocol can be used in tandem with a low performance chip. However, the scalability and the extensibility of the product would be hampered if this is the choice made. Regardless of the power of the processor, MQTT is the baseline protocol that is planned to be used as MQTT does not have an upper bound for performance requirements. The protocol is single threaded but programs can be written to make it thread-friendly. However, threading is not required for this project because the communication only occurs between the module and the mobile device.

#### 3.6.2.2 Extensible Messaging and Presence Protocol

Extensible Messaging and Presence Protocol (or XMPP) is an application layer protocol designed for messaging as the name suggests [26]. The extensibility of this protocol allows it to function beyond what it was designed for.

Like the MQTT protocol, XMPP is also designed to be simple. The focus is on sending messages as XML content. HTML, a subset of XML, is commonly used with HTTP however. The distinguishing factor, however, is the decentralized nature of this protocol. This does not affect the project though as the mobile device and the module are supposed to be two independent systems communicating with each other and acting based on their current states.

XMPP offers near-real time communication. But this is somewhat misleading as it does not mean instantaneous. It is relatively slower due to the decentralized nature but that does not hamper it from being a good option for this project.

#### 3.6.2.3 HyperText Transfer Protocol

The HyperText Transfer Protocol or HTTP is one of the most widely used protocols on computer devices. The HTTP is rather a simple protocol in design but with many options [27]. However, it is not a communication protocol in the sense that it is not really designed to be used to send messages among two computers. However, the popularity of APIs such as REST or SOAP has led to its use in message passing as well. The overhead is somewhat higher in comparison to the MQTT protocol but the use of API on the web is nothing but common.

An advantage HTTP has over MQTT is the ability to choose if the connections have to be kept alive or not. However, for his product, the connection must be alive and persistent for optimal interactivity with the mobile application.

HTTPS is an extension to the HTTP protocol which adds TLS (transport layer security) protocol to the HTTP protocol. TLS is an application layer security protocol that uses a public key and a private key system to ensure that the data transmitted are from the original senders and also to prevent snooping and spoofing in order to keep the level of privacy high.

## 3.7 Peripheral Communication

Various sensors, buttons and other components of this project would use specialized hardware, simple or complex, to communicate with each other by transmitting or receiving information. There are various communication protocols, standardized or non-standard, free or proprietary. Most microcontrollers support these protocols and especially the ones listed in this document. Some of the common protocols are used are UART, I2C, SPI, etc. Each of them have their own advantages and their disadvantages are listed in the following sections.

### 3.7.1 Universal Asynchronous Receiver Transmitter

Universal Asynchronous Receiver Transmitter (or UART) is a well known, commonly used, cheaply implementable protocol. The data is serially transmitted back and forth through separate lines,

or on the same line, making it simplex, fully-duplex or half duplex depending on the usage. The lowest order bits are the first ones on the wire making it similar to many of the network layer protocols used. One of the prerequisite for using UART is knowing the baud rate of the transmission by both the transmitter and the receiver, mainly because the protocol does not have a clock line like the other protocols in this section. However, more expensive versions of the hardware that support UART allow automatic detection of baud rate.

### 3.7.2 Serial Peripheral Interface

Serial Peripheral Interface (or SPI) is a communication protocol, originally introduced by Motorola, is a synchronous, full-duplex communication protocol. It extensively uses shift registers to store values and as the name implies, it uses a serial model. Unlike UART, SPI uses a clock line to synchronize data communication. SPI uses a master-slave architecture, which means there is a device that instructs other devices in the SPI network. While it is possible to have multiple masters, SPI is typically only used with one. Some of the uses for SPI involve working with displays and storage devices, like EEPROMS and SD cards. SPI is a four-wire serial bus, however, not all of them have to be used. The chip select line and the two I/O lines can be omitted depending on how the data is used. If there is only one slave device, then chip select can be set to high. Master-in Slave-out (MISO) can be ignore for output devices like display and Master-out Slave-in (MOSI) can be ignored for input devices. However, when using multiple slave devices, the lines are important and the way the MISOs and MOSIs are connected makes a big difference.

### 3.7.3 Inter-Integrated Circuit

Inter-Integrated Circuit (or $I^2C$ or I2C) is a half-duplex communication protocol, patented by (then-Philips) NXP Semiconductors. It has two lines, a SDA, the data line, and SCL, the clock. I2C is used with various low-speed devices like sensors, digital electronics components etc. I2C also uses the master-slave (or controller/target as the latest specification document calls it) like the SPI. Unlike SPI, I2C works with fewer lines, but at the cost of lower speeds. I2C uses an address system to work with multiple slave devices, effectively using a traditional bus system to communicate with multiple devices with just two lines. This however comes at the cost of higher overhead. Figure 3.7.3.0.1 gives a basic yet a clearer picture on this overhead.

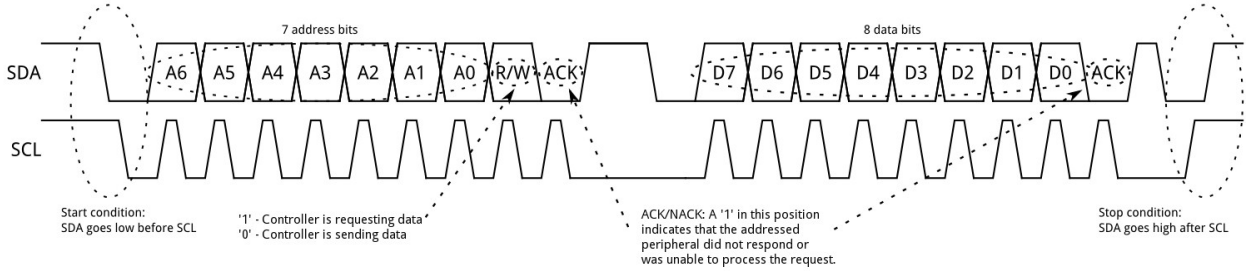



Figure 3.7.3.0.1: Typical I2C Data Line Signal (Sparkfun CC 4.0)

## 3.8 Anti Theft Sensor

The sensors are needed for this project design to be able to trigger the anti-theft alarm. There are a variety of sensors that can be chosen from. However, there are two types of motion sensors that can be feasibly implemented for the purpose of triggering the alarm.

### 3.8.1 Motion Sensors

Motion sensors, as the name suggests, are electronic devices designed to sense movement of anything ranging from people, parts, to any object. Specifications such as the sensor minimum and maximum speed as well as the distance range need to be identified to help choose the correct motion sensor for the anti-theft security system. Motion detectors are generally used for applications that involve automatic door openers and street lights or even in burglar alarm systems[28]. Although there are a variety of sensors, there are also a range of motion sensors that fit certain applications. By doing basic research on these types of motion sensors the team will be able to identify which out of the options to move forward with for our security system needs. It's also important to note that not all motion sensors will be discussed for the purpose of avoiding unneeded types.

#### 3.8.1.1 Active Detectors

The first type of motion sensor is an active detector, which are radar based motion sensors that use radio frequency waves to sweep the area around it until an object is hit. Once the object is hit, the radio wave bounces back to the sensor detector. After the radio wave is retrieved, the motion sensor will emit another wave and detect whether the object moved based on the frequency shift once the wave is retrieved again. Figure 3.8.1.1.1 demonstrates how an active motion detector functions.

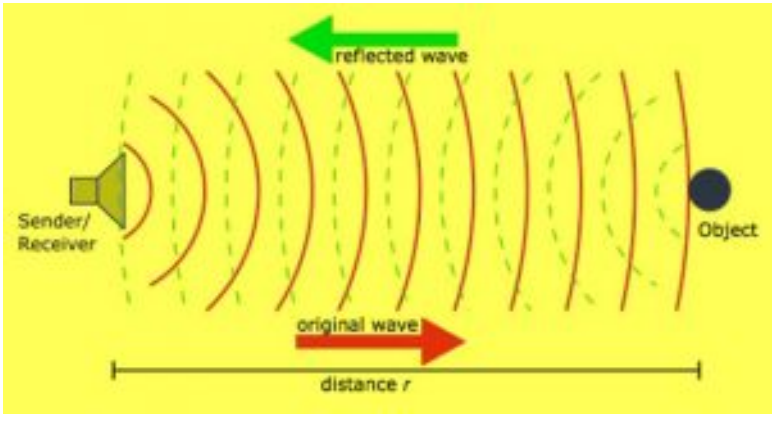



Figure 3.8.1.1.1: Demonstration of how an active motion sensor operates

Although this sounds great for what the anti-theft system seeks, the biggest issue with this technology is the sensitivity to a variety of size objects that move in front of it. Assuming a bike is parked outside in the public, many pedestrians could happen to walk in front of the sensor

and trigger it. We want the sensor to only trigger upon a small distance as well as the condition of detecting it is a human. Which is definitely possible, but not through active detectors, thus this option cannot be used unless we want our anti-theft system to have a high chance of being defective and inefficient [29].

#### 3.8.1.2 Passive Infrared Sensors

Passive infrared sensors, also referred to as PIR sensors, detect skin temperature. It functions through emitting black-body radiation at mid-infrared wavelengths [28]. The distinction is made through the background objects at room temperature. The ultimate purpose of this type of motion sensor is for detection of objects, people, and even animations infrared radiation. PIR sensors can also be adjusted through the utilization of potentiometers, which can alter the sensitivity as well the length of a PIR activation once triggered [30].

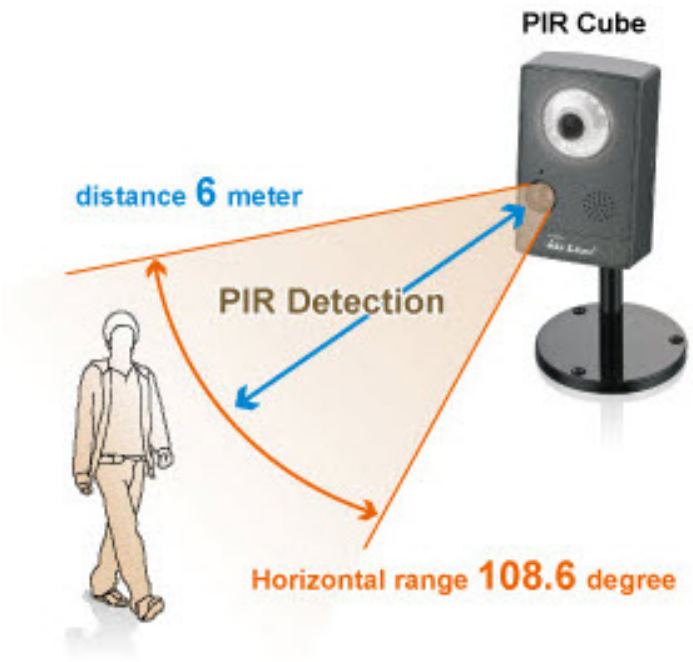



Figure 3.8.1.2.1: Demonstration of a PIR sensor [31]

A passive infrared sensor example that the team is considering utilizing for the anti-theft system is the SS-430L-BK found at Digikey. The supplier is Kemet and the characteristics of the PIR is shown in Table [X].

### 3.8.2 Vibration Sensor

Vibration sensors, also referred to as piezoelectric sensors, is a potential option since it can detect a thief through measuring the amount and frequency of a vibration detected on the bike. The vibration sensor will involve a use configuration option so that upon the type of sensitivity level the sensor can send a signal to the microcontroller to set off the Anti Theft Alarm.

A great candidate that can be used as the Anti Theft sensor is the LDTM-028K, shown in Fig. 3.8.2.1. This product is sufficient because depending on its assembly its application can be used as an accelerometer or vibration sensor. The vibration sensor will be used as previously discussed meanwhile the accelerometer is an option that can be applied for further detection of bike theft.


Figure 3.8.2.1: The LDT0-028K

The LDT0-028K datasheet[32] also provides the sensitivity of the vibration sensor based on the effect of added mass shown in Table 3.8.2.1.

Table 3.8.2.1: LDT0-028K Vibration Sensor Sensitivity Based on Added Mass

| Added Mass | Baseline Sensitivity | Sensitivity at Resonance | Resonant Frequency | +3 Db Frequency |
|---|---|---|---|---|
| 0 | 50 mV/g | 1.4 V/g | 180 Hz | 90 Hz |
| 1 | 200 mV/g | 4 V/g | 90 Hz | 45 Hz |
| 2 | 400 mV/g | 8 V/g | 60 Hz | 30 Hz |
| 3 | 800 mV/g | 16 V/g | 40 Hz | 20 Hz |

## 3.9 Mobile Application

The mobile application is an optional utility that could be used to improve the user experience when it comes to using the SAFE T module. A use case would be not having to get close to the bike on which the module is hosted to arm the alarm system. Instead, the mobile application could be used with range to have that done. There are many other use cases which are detailed

in this document. In this section, many third-party entities are mentioned or discussed. The team behind this project is neither affiliated with them nor are they endorsed, and vica versa.

### 3.9.1 Operating Systems

Our intent with the Mobile Application is to create an interface from the user's mobile phone to the product in order to provide a more dynamic and compacted system controller. The application is intended to be supported on both iOS and Android platforms, which will pose a greater challenge but allow for greater accessibility.

The two largest mobile phone operating systems are Android and iOS as shown in Figure 3.9.1.1. For this reason, we have chosen to support the two operating systems as their collective user share is over 95

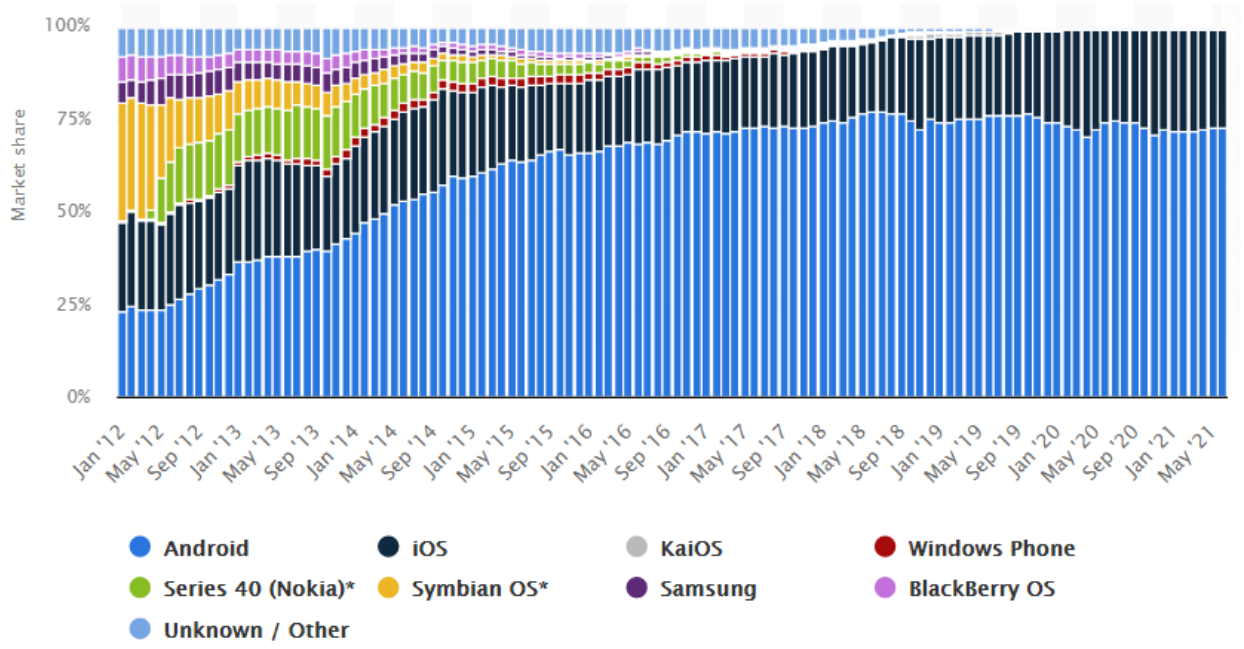



Figure 3.9.1.1: Worldwide Mobile OS Market Share [33]

The following section offers different options available to write applications for these platforms and their major benefits and trade-offs.

### 3.9.2 Application Platforms

Both Android and iOS have tools to develop applications and have extensive documentation by their developers and third-parties to develop applications for the platforms. However, the platforms use different technologies that makes developing code for them difficult, especially for a small programming team. The following subsections contain more detail on why it is not feasible for a small team to have native codebases for the mobile applications.

#### 3.9.2.1 Native Android Application

Android is an open-source operating system by Google, written based on the widely popular and free and open-source Linux kernel. Being the most used operating system, even when considering desktop and server platforms, the amount of documentation and size of support community that Android has is unparalleled.

Using Android Studio and Software Development Kits for Android, the native application to control the product can be written with the least amount of compilation and execution overhead. The languages supported by them are Java and Kotlin (Kotlin is designed to be interoperable with Java), as the operating system model is built around having a run-time environment called Android Runtime Environment (ART), similar in design to the Java Virtual Machine.

Google also provides a development kit called Native Development Kit (NDK) that allows parts of the code to be written in C or C++. However, they are not required as the application will not be using performance critical third-party libraries.

#### 3.9.2.2 Native iOS Application

iOS is an operating system by Apple Inc., which has a BSD-like design. The platform is closed and restrictive with respect to the device on which it runs. However, this does not mean that other devices cannot be interacted with using the platform.

Apple provides the first-party tools, software development kits required to develop applications for iOS using Objective-C or Swift. These tools are supported by the development environment Apple provides called Xcode, which is used to also write software for other Apple devices and operating systems. One benefit that Objective-C brings forth is the ability to utilize the existing C and C++ libraries, without the need for additional SDKs, like Android's NDK.

### 3.9.3 Multi-platform Software Development Tools

The fundamental goal of the multi-platform software development tools is to provide an abstracted, easy to use platform to create mobile applications without taking too much overhead, while providing the best possible user experience. For a small time with a limited amount of time, this is the most viable option. But choosing the right framework or the right development kit can prove to be difficult. Since our range of operating systems supported is small, we have a large number of options for creating an application.

However, for this project, wireless connectivity is one of the major requirements and if the framework cannot reliably do so, then it will not be considered. The stretch goals requirements will also impact the decision which means the better the framework is suited for the project, the higher the likelihood of choosing it for this project.

There are a lot of options for developing both Android and iOS applications with some of the top contenders being React Native, Xamarin and Unity.

#### 3.9.3.1 React Native

React Native is User Interface Software Development Framework for developing applications that run natively on the operating system unlike the React framework, which is used for creating web UI components. The framework is easy to use if a developer has worked with the React framework before. The two frameworks, by Facebook, use the same philosophy toward building applications for mobile devices (and other platforms out of the scope of the project) except when it comes to UI manipulation. In fact, React Native can be considered a subset of the React Framework.

The React framework uses a virtual Document Object Model to manipulate the HTML Document Object Model but native applications work differently. The React Native framework also uses JavaScript, which also allows the use of TypeScript (a JavaScript standard with support for data type for better code quality) for writing applications. The JavaScript code is run on the machine and uses a bridge communication system with the operating system to achieve actions only possible by native software. This means the UI components are not emulated. However, there is a small overhead to having JavaScript interpreted but that is the trade-off for multi-platform tools. Additionally, the framework also allows running platform-specific code to allow programmers to do things that are platform-specific.

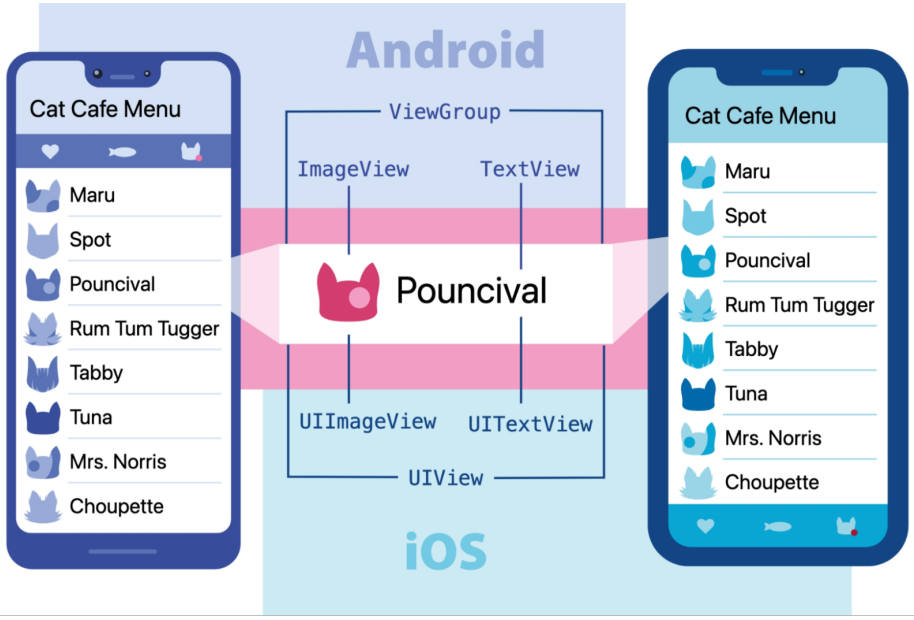



Figure 3.9.3.1.1: React Native User Interface Components on Android and iOS devices (CC 4.0) [34]

As shown in Figure 3.9.3.1.1, React abstracts the native UI components to the closest equivalent on the target platform. For example, the React Native component <Text> corresponds to the <TextView> component in Android and <UITextView> component in iOS in the figure. Also, notice how they have different placement and styling even though they were written with the same codebase.

The modular approach to application construction and React's specificity on not doing many

things in one program means that the native operating system calls or the wrappers for them are not provided by React Native. Instead, either the platform-specific code is to be used or a third-party library is to be used to get the required functionality. Since JavaScript is a popular language, React Native can be used to easily create the applications for this project while only having to focus on the required native calls for the location and Bluetooth data.

#### 3.9.3.2 Flutter

The goals of the Flutter User Interface Software Development Kit lies strictly in making fast, cross-platform applications with a single codebase. This platform by Google focuses on UI development as the name suggests has more of a focus toward writing applications with good user interface.

Flutter uses the Dart programming language by Google, which compiles or sometimes transpiles to other languages based on need and the software environment to ensure high performance. However, the goal of this project is getting the application to work with the native operating system calls that lets us fetch location and bluetooth data, which Flutter does not support out of the box. However, there are third-party libraries for Flutter from pub.dev, the official package repository for Flutter and Dart.

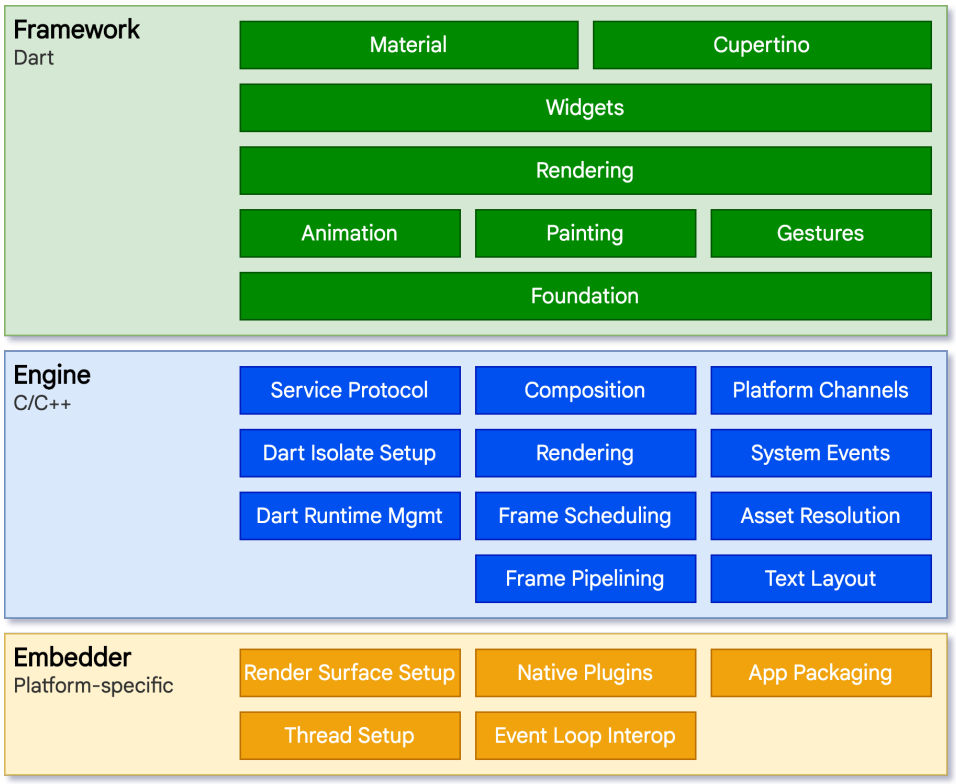



Figure 3.9.3.2.1: Flutter User Interface Software Development Framework Architecture (CC 4.0) [35]

Flutter does not use the native user interface components for applications. Instead, it emulates the User Interface of the target platform using the rendered in the Flutter Engine (Figure 3.9.3.2.1).

The reason for doing this is to make it easier to have Flutter layout the UI to avoid errors. The team does not have prior exposure to the Dart programming language but the similarities with JavaScript and other object-oriented programming languages make it a good candidate for this project.

#### 3.9.3.3 Apache Cordova

Cordova is an application development framework by the Apache Foundation that lets developers use web technologies: HTML, CSS, JavaScript, to build mobile applications. This also includes support for the underlying hardware of mobile devices, which means it supports operating system calls out of the box. However, the documentation for this framework is not extensive and the project is not frequently updated by the developers.

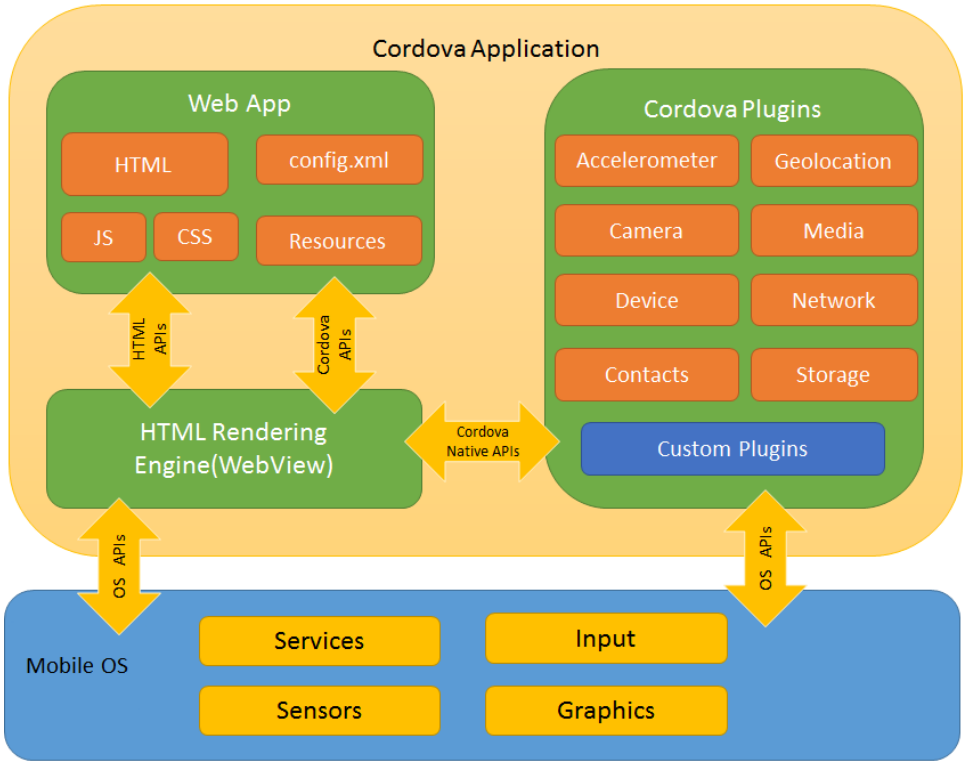



Figure 3.9.3.3.1: Cordova Platform Architecture (Apache License) [36]

What makes Cordova a good candidate is the relative ease of use and the simplistic architecture that utilizes a web renderer and plugins with Cordova API for operating system calls and access (Figure 3.9.3.3.1) . But the slow development of the framework itself and lack of popularity which leads to fewer documentation and third-party support makes it more difficult to choose this as the framework for this project.

#### 3.9.3.4 Haxe

Haxe is a cross-platform programming language and compiler used to create source code for multiple platforms from the same codebase. The compilations do not, however, specifically

target a device rather language platforms or virtual machines. The C++ target is what is used to run applications on a mobile platform. Using Haxe requires a deeper understanding of other platforms on which Haxe code runs, which unfortunately makes it more difficult to consider this software.

#### 3.9.3.5 Xamarin

Xamarin is a software development platform that extends the .NET platform [37]. .NET is a platform by Microsoft that uses the Common Language Runtime as its runtime environment, similar to how the Java Virtual Machine works. Similar to the Java standard library, the .NET platform also provides a wide-range of libraries to create applications. Since .NET used to be exclusive to Microsoft platforms, Xamarin was created as an extension to support the C# language and the .NET language on non-Microsoft platforms. However, Xamarin was acquired by Microsoft and now is part of the official Visual Studio toolset and .NET tools used to create applications.

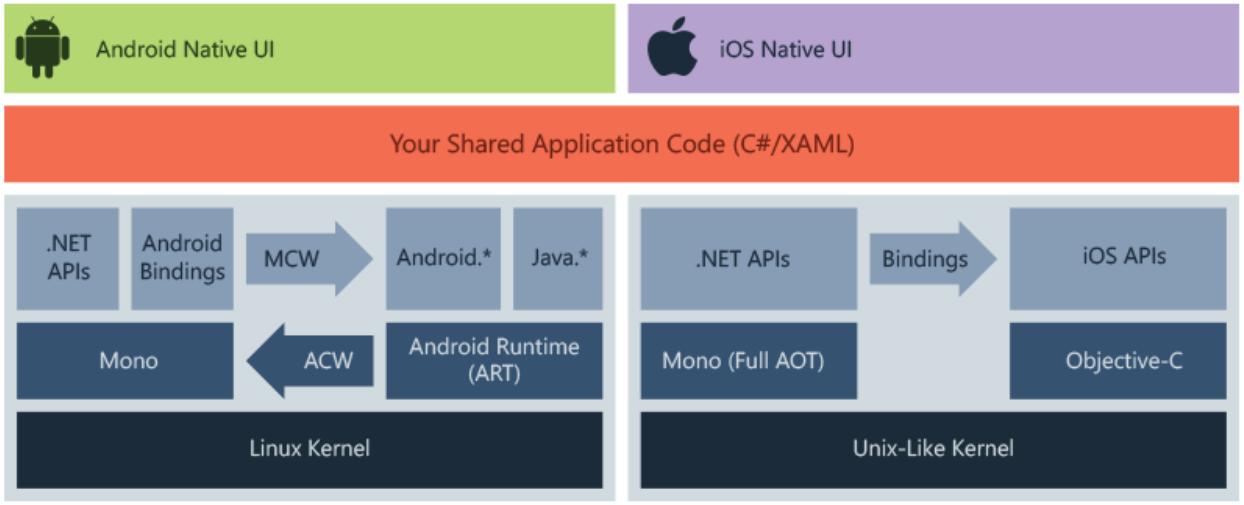



Figure 3.9.3.5.1: Xamarin Architecture (CC 4.0) [38]

Xamarin provides libraries to create applications on iOS and Android (and other devices) with native user interfaces while having the same codebase (See Figure 3.9.3.5.1). However, Xamarin is not as popular for creating applications and has less extensive documentation compared to React Native or Flutter. Moreover, the team is not experienced in C# (or F#) to create .NET applications and in XAML to create UI, making it a somewhat less preferable alternative.

### 3.9.4 Development Environments

Development environments dictate where the program is tested, compiled, run, and debugged. While the testing may not be as accurate as running the application on the targeted machines, these environments provide various tools like simulators and/or emulators, code analysis tools, etc. to ensure that the application built is less likely to fail or break compared to not using those tools.

#### 3.9.4.1 Visual Studio

Visual Studio is one of the most used IDEs developed by Microsoft. While it is primarily used to develop and work with Microsoft products in C++ (on Windows) and Microsoft's C#, it allows for a lot more support. The dotnet framework is also primarily developed on Visual Studio, making it ideal for working with Xamarin.

#### 3.9.4.2 Visual Studio Code

Visual Studio Code is a separate tool from Visual Studio by Microsoft with a lot of the features that are available in Visual Studio like IntelliSense, a code completion tool. While a text editor, this program does not require much to make it a competent editing software with near IDE like features. The ability to install extensions, just like Visual Studio, makes it a software capable of improving developer productivity. React applications can be easily developed on this software, and various debugging extensions for a wide variety of languages and tools are available.

#### 3.9.4.3 Android Studio

Android Studio, as the name implies, is an integrated development environment by Google to develop native applications for Android. It is based on the Java and Kotlin IDE called IntelliJ IDEA, by JetBrains the company behind the development of the Kotlin programming language, is one of the most commonly used IDEs for Java development. Google also supports Android Studio for developing Flutter applications.

#### 3.9.4.4 Xcode

Xcode is the development environment that Apple provides for developing software on their devices. They primarily support Objective-C and Swift. They also come with simulators for Apple devices and debug tools which are necessary even when using third-party tools to develop applications.

### 3.9.5 Build Tools

Many of the tools required to build software come packaged with the vendor's integrated development environment. However, this is not always the case and developers have to specifically choose what they require to successfully build applications.

#### 3.9.5.1 Expo

Expo is a creation, execution and deployment tool for React Native projects which simplifies the process of using React Native, and is used in the official documentation for React Native. While React Native simplifies many things for multi-platform application development, there are so many things involved in setting up the software, building for each device, and deploying them to a physical device for testing.

### 3.9.6 Deployment and Publishing

Deploying and distributing applications is a key part of mobile development. This determines how the users of the product get access to one of the key yet optional parts of the project. Making it difficult to acquire the application makes for a poor choice from the developers' part. Hence it is the responsibility of the developer to take the steps to ensure the users have access to the application without major effort.

The two operating systems supported come with application managers that allow for easy installations and updates of software. However they are tied to stores that are vetted.

#### 3.9.6.1 Android Packages

Deploying applications on Android is easier than on iOS as Android is an open operating system and iOS is extremely closed. Android allows installation of applications without having to use the package manager the operating system usually comes with: Google Play Store. Instead, external files can be used to install packaged applications, specifically, the Android Package Format (.APK files), which is similar to the Java Package files (.JAR) in structure. This is not surprising as Android heavily draws from the design of the Java Virtual Machine (JVM), so much so that it used to have the Dalvik virtual machine, a virtual machine which was very similar to the JVM, before being phased out in favor of ART.

#### 3.9.6.2 Google Play Store

Google Play Store is the most used application distribution platform on Android. To do so, a one-time developer fee is required to sign-up for a Google Developer Account. However, that is not where the publication requirements end. The developers are also supposed to create End Users License Agreement (EULA), licensing information, Terms and Conditions, Warranty information, and Privacy Policy on the legal end. The technical end requires signing the application using a digital key for verification and to prevent spoofing, application logo files, application screenshots, video trailer, application tagging, age rating, and pricing. Doing this all allows the developers to have the application reviewed by Play Store, which would then allow the application to be listed on the platform.

#### 3.9.6.3 Apple App Store

iOS, unlike Android, has a heavily closed system when it comes deploying applications. Applications cannot be packaged and installed outside the Apple's App Store. Even for debugging during development, the developer must have a macOS (required for iOS development as the SDKs are not available on other platforms) connected to their iOS device. The process for publishing the application is similar to Google Play's system. However, anecdotal reports suggest that Apple's review system is far stricter than Google's, keeping it consistent with the entire design of the iOS platform.

### 3.9.7 Telemetry

Collecting usable data from the module is also an important task that the mobile application must accomplish. However, the data must be written to a location where it can be easily and exclusively accessed only by the application owner. One way to ensure this is having an authentication system thoat would store all user data on a server. However, this is not feasible as the scope of the project does not allow the implementation of such features. Instead, simpler solutions are preferred.

One of the easiest options is to set up a Sqlite database within the application platform that could contain all the user data. This database file will not leave the user's operating system. Alternatively, the Operating System APIs can be used to store app data, in the space specifically allocated for the application by the operating system. An alternative would be using the cloud storage services that come with the operating systems: Google Drive for Android and iCloud for iOS to have the data backed up and accessible across all devices. However, this may prove difficult to implement as operating system and cloud service specific code are required to get them functioning correctly. If an operating system supports having the allocated space be backed up, then that option may be preferred.

# 4 Design Constraints

Researching and carefully planning out how to theoretically design the SAFE T project sounds all well on paper, however, in terms of real-life there may be considerable issues that may hinder the project. These constraints are bound to happen in regards to the process of developing a final product, thus these constraints need to be addressed and analyzed to help reduce any complications that may occur in the last stages of development. Moreover, the constraints can also lead to a domino effect, where a constraint can lead to another, resulting in a highly damaging chain reaction. So, it is crucial that the team and the product are not restricted by these constraints and every effort to minimize the likelihood of them happening will be made, in addition to the resolution tactics mentioned above.

## 4.1 Economic and Time Constraints

The economic constraints to a completely "ground-up" method of building this design would be consuming; a smartphone and a bicycle are required for proper assembly. However, there are multiple approaches to this hurdle. One would be to assume that in the current modern world, a smartphone is accessible to every person and is owned, whether it be an older or a newer model, by over 80% of the population. This number does not factor in age, and it can be assumed that an older individual who may not own a smartphone would not be physically able to ride a bicycle at commuter-levels. The bicycle is a less-avoidable constraint, as bicycles sold as new can vary from hundreds to thousands of dollars. The researchers and designers are using one member's bicycle, this member being a commuter cyclist, which allows both access to a bicycle and adds perspectives that can positively influence the design. Considering these two large economic hurdles covered, the other economic constraints are virtually zero; any other expenses of electronics including the microcontroller will be far below $300. Split between four (4) designers comes at most $75 per-person, which for a design project is considered reasonable. On the consumer side, the economic constraints would be similar to the economic constraints of the designer; this product is not intended to be used by persons without both a smartphone and a bicycle; making the potential cost of the design small in comparison.

The time constraints of this design revolve around the date of 'delivery'. This date, which is April 15th of 2022, is non-negotiable and unable to be moved back beyond what has already been set. Similarly, the design prospects need to be nearly completed, or set in a position to be completed, by December 1st of 2021. The current design process is split into two sections: theoretical, and practical; the theoretical stage is in motion to be completed by December 1st, 2021, which will include all the blueprints and design choices that are needed to move to the next stage, which begins after December 1st, 2021. The practical stage is where the pieces are assembled and constructed following the blueprints and models, with tweaks and troubleshooting updating the blueprints as needed (for this reason, the two design stages are NOT mutually exclusive). In order to combat and overcome these time constraints, a "milestone" table is part of the early design

stages, and covers the general parts of both the theoretical and practical design, with appropriate due dates set. See section 9.2 for a detailed description of the milestone table.

## 4.2 Environmental, Social, and Political Constraints

In terms of this project and its relation to any environmental, social, and political constraints there are a few the team will consider for the project design. Our priority out of all three types would be social. The ultimate goal of this product is to benefit bicycle riders in regards to safety, cost, and ease of use. The next priority would be environmental constraints such as weather conditions, energy consumption, and components needed. Political constraints don't exist at this stage of development but can exist later in the development stage.

Social constraints as mentioned involve safety, cost, and user-friendliness. These factors will greatly impact whether a customer would be interested in our SAFE T design. Addressing safety, we wish to implement all lights in the lighting system to provide sufficient brightness and light angle visibility to protect bicycle riders from being harmed by accidental car crashes. Cost is the next factor the team wishes to consider in the product design to increase interest and affordability. Affordability can be addressed through means of analyzing the cost of all the materials needed to produce the project product. Once the cost of all materials is calculated, the team can do further investigations to compare our product to similar product competitors to influence our decision on the cost of the product for commercialization. The last factor is user-friendliness, this matters as much as the other factors in such that if the product does not possess this quality then customers may lose interest in our product.

User-friendliness needs to be considered in regards to the installation of all equipment as well as the pairing of the phone app for light control. This factor will be addressed by reducing the complexity of any wire harnesses to avoid confusion upon installation, making all equipment strong and steady to decrease chances of falling, and allowing the phone app to be easily pairable with the lighting system of our product. By addressing all social constraints, our project can have a higher chance of having customers like our product. Environmental constraints consist of energy consumption, air, water, or some form of pollution. The project will be waterproof and safe to use in all weather conditions to preserve the protection of the customer. Air pollution will be minimal through the use of small electronics that require small power. Adding on, energy-saving options will be utilized to help preserve the power of the product, which will in return help bicycle riders use our product longer without concern of short battery life. The battery charging the product system is aimed to be rechargeable to further avoid any battery usage. This product will ultimately strive to be environmentally friendly.

The project is also legally constrained by various environmental laws of various regions. The use of chemicals that are toxic for the environment are avoided. Even for the components that are used, the environment safety certifications are considered a plus even if it involves minor increases to the project budget. The team will try to the greatest extent to adhere to the Restriction of Hazardous Substances (RoHS) directive also known as the lead-free directive, which is the standard toxic waste restriction directive. Even though it is a European Union directive, it is widely recognized internationally and companies try to adhere to this standard. This means that the use of products

containing lead, mercury, cadmium, hexavalent chromium, polybrominated biphenyls (PBB) and polybrominated diphenyl ethers (PBDE) in electrical and electronic equipment are restricted [39].

## 4.3 Ethical, Health, and Safety Constraints

In terms of any ethical, health, and safety constraints this project may have, the team prioritizes addressing health and safety. Health is an aspect due to the product lighting system's brightness in such that if the brightness is too bright it may hinder those around the customer in regards to vision. This is dangerous because if, say, the taillight is constantly emitting too much brightness then drivers behind the bicycle rider could potentially get distracted. Vision is incredibly crucial in driving and the product serves to only highlight visibility to the bike. A slight distraction for drivers could cause a car accident or some form of health damage. So, adjustments to the LED brightness of the lighting system will be addressed to keep the bike rider and those around the rider safe.

Besides concerns of vision blockage from our product as an obstacle, which will be addressed, another health constraint for the design process is COVID-19. The virus still persists and there are various restrictions the University of Central Florida have implemented to preserve protection of the students. The restrictions placed, although of course prevents students from catching the virus, is also an obstacle in a sense that labs are restricted, which means less equipment availability. COVID-19, although a big obstacle to the project design process, still can be addressed through online communication between project members and careful scheduling. The project is still feasible and can be accomplished.

Another safety concern is the use of a mobile app used during operation of a bicycle can be considered a distraction and is therefore dangerous to the operator and to people surrounding the operator (both vehicles and pedestrians). In general, this problem as a whole is unavoidable due to the nature of the project; it is impossible to make an app for bicyclists to use during riding that does not pose, at least in the very slightest, a minor visual distraction. For this, it is considered the risk of the user to always be alert of their surroundings and upcoming environment before looking away, even briefly, to look at their phone (or anything for that matter). As mentioned, this safety concern is a risk that must be considered and accounted for by the user. This can be related to the use of apps that are designed to aid drivers; map and directional programs are used by drivers with the intention that they are not used as a distraction, but rather placed in such a way that they can be viewed without impairing the drivers ability to operate their vehicle. Due to the justification of this safety constraint, it can be considered an ethical constraint as well; the designers have to assume that the users of this project must account for their own safety. With the inclusion of an assumption of risk warning, the ethics surrounding the use of this device are common to modern life where smartphones accompany the actions of nearly everything. As mentioned in section 1, the purpose of the project is to decrease the injuries and fatalities attributed to cyclists; the possibility that someone could be injured due to this project exists, but is not being actively accounted for.

## 4.4 Manufacturability and Sustainability Constraints

The manufacturing constraints are tied into the design specification (see section 8). Every major component of electronics will be pre-manufactured (i.e., bought), making the manufacturing stages of all electronic equipment not on the shoulders of the designers. The electronic components, both for the microcontrollers and the more basic parts (e.g., the buttons/switches) are too complex for manufacturing from scratch by the designers. For modern designers, any embedded system, even the most 'simple' (i.e., user-friendly) would be manufactured by an electronics company such as Texas Instrument. With the electronics portion covered, the two other portions of manufacturing lie in building parts for the container (chassis; see section 8.3), and securing parts to the bicycle. The first portion, which revolves around building housing for the electronic components, will be carried out using CAD software and 3D printed from common printing plastics. The manufacturing of the parts will first involve measuring the absolute maximum dimensions that can be taken up by the microcontroller house; this is the only part of the design that can cause potential problems in design as making it too large in any dimension can cause problems for the user (e.g., being too wide or two tall can interfere with bicycle operation, which causes more problems for the cyclist than the project helps). Once these dimensions are established, the general size of the chassis can be determined by measuring all of the electronic equipment that will be held inside (that is, the microcontrollers). Ideally, there should be as little difference in the inside dimensions and the microcontrollers as possible in order to reduce any shifting of the electronics. Also, if needed, the microcontroller may be rotated to better accommodate the dimensions of the area that the chassis will sit on; this is a design stage that can be troubleshooted in later development. The chassis will be manufactured in two major pieces; one that connects to the frame, and another that shields the electronics; the former can be ideally designed and printed once while the latter may need adjustments along the stages of the design process; if this occurs, the shield may be reprinted with new dimensions or it may be adjusted with simpler 'padding' techniques (e.g., using duct-tape to add extra dimensions of a few millimeters if needed). The 3D printing may cause a design constraint, as the acquisition of 3D printing services (both retail and those provided to students) can be time-consuming; because of this, the chassis and shield pieces should be built on the CAD software as close as possible and 'padded' if necessary first before returning to multiple print interactions. Other than the printing, the design requires the securing of the buttons and the wires to the handlebars of the bicycle. These can be secured using duct tape or electrical tape to begin, and as design becomes finalized can be secured with other methods that may include additional printed CAD models or more simple metal joinery pieces. One overall constraint of the manufacturing process is that the designers are unable to extensively analyze the mechanical factors that may go into the strength of the product. This means that the CAD design will have to be done using general understandings of sustainability; meaning that the design process will not involve any formulas related to mechanical engineering. The design will be done in such a way that the model should be well beyond robust by any visual standard.

Sustainability is a large part of the non-electrical design. Bicycles and cyclists are commonly caught in rain, be it expected or unexpected, and as such all exposed electrical pieces must be protected from the rain. Virtually any and all 'outside' conditions that can affect a person or a motor vehicle can affect a cyclist and affect this design; sunshine, dirt, rain, wind, etc. However,

rain and water are the only things on that list that can essentially "brick" the design (that is, render it completely useless, inoperable, and unfixable). Because of this, protection from the rain is the top sustainability consideration. The plastic box (chassis) that the microcontroller will sit in will be manufactured in such a way that any reasonable amount of weather and rain will not come into contact with the electronics.

As discussed in section 8.3, the microcontroller holder will need entrance and exit ports for the wires that will run to the buttons and to the screens; to minimize (and ideally eliminate) water damage, these holes will be as close to the dimensions of the wires as possible, and placed on the bottom so that rainfall had less of a chance of seeping in. Along with this, the shield itself on the inside will be padded with a water-resistant coating that should dissuade water from potentially seeping its way up the wires. If designed and built adequately, the wire holes should be the only potential egress of water for the chassis. As mentioned in the manufacturing constraints section, the box that the microscontroller sits in will be as close to the size of the microcontroller to eliminate sway inside the box. Sensitive electronics such as embedded systems can not be expected to take rough jostling that other cyclist accessories could take (e.g., a water bottle holder or a bell). Also mentioned in section 8, the largest sustainability constraint is that a cyclist can be hit by a motor vehicle or fall in a high speed crash. In this occurrence, depending on the strength of the impact, the bicycle and cyclist can become seriously damaged/hurt. In these occurrences, the health and well-being of the cyclist should take first and major priority. However, in low-speed crashes that can occur due to minor accidents, a cyclist may be rendered stopped and/or startled, but not injured, with their bicycle potentially thrown to the ground. In occurrences like these, any component that does not make direct contact with concrete or packed earth should be expected to still operate, if maybe become scuffed. As with keeping out the rain, the microcontroller box should be built so that small jostling will not break (or brick) the microcontroller.

Another design constraint is tied to the versatility of connection with the plastic shell that will hold the microcontroller to the bicycle (refer to section 8). The design of the plastic parts that will be attached to the bicycle to hold the microcontroller in place (referred to as the "chassis" in section 8) are designed in mind of the bicycle in possession of the designers. However, this bicycle is a "road bike" and can be considered: (1) to have the frame shape and dimensions of common road bikes (that is, nothing in the bicycles design can be considered abnormal to common road bikes) and (2) to be the style and size of the bicycles that would most benefit the use of safety lighting (refer to section 1.3). This being stated, the design of the chassis that conforms to the frame (i.e. "tubes") of the bike are set to fit the designer's bicycle (dimensions given in section 8). Ideally, these pieces would contain an adjuster that would make the chassis more universal to frame dimensions of most bikes, however this attachment goes beyond the scope of the design (i.e., would require more advanced mechanical knowledge that the designers do not currently possess). In order to mitigate this constraint, the design will not feature an adjuster, and will instead be 'single-sized'. One thing to consider is that while the chassis design may not be universally sized, the focus of this project lies in the electronic hardware and in the software, which will be universal for any user of this product. While the lack of adjustable parts may be a design constraint, it could be fixed well after the initial prototypes have been completed with no foreseeable change in electronic hardware or software.

## 4.5 Standards

This section will cover the various standards which relate to our product. Due to the need for consistency when working in concert with others, end-user concerns and lack of technical know-how along with the inherent desire to decrease cost, standards were developed. The establishment of standards provides us with guidelines which we must follow in order to provide safety assurance to our customers and meet governmental regulations. Each section below will refer to a number of standard-making bodies and though some are different, there are a few staples that manufacturers and product makers rely on. Some of the most prevalent are as expected, IEEE, IEC, ANSI, and UL along with a variety of relevant government entities.

### 4.5.1 Battery Standards

Due to the battery's importance in our system, it is more-so important that we establish the necessary standards and follow them accordingly. Without proper adherence to the following standards, our product will not only be more difficult to transport and sell, it will be generally more dangerous. Lithium-ion batteries are known for their potential for combustion when mishandled, especially when dealing with charging and discharging. In order to fully protect our product and our consumers, we must thus strictly adhere to some very common standards. With greater availability of Li-ion powered devices, the implementation of these standards increases and gives us a great idea of what we need to work up towards by providing a multitude of examples [40] [41] [42]. Though many standards are not legally enforceable, they represent a general consensus of what is proper form and technique, these voluntary standards are governed by varying bodies but these bodies generally form a consensus with relevant manufacturers in order to establish the standards, this means that the governing body is not far-removed from the implementation and actually listens directly to those whom it would affect. Standards are stable, they do not change very often and changes are generally representative of a paradigm shift within the respective field or market. In the case of batteries, the standards very rarely change, usually it is due to a new or updated technology.

There are a variety of Standard-making Bodies which provide product creators with extensive guidelines. Some bodies involved in battery standards include IEEE, ANSI, IEC, UL, and the US DOT which unlike the rest, maintains legal authority and may enforce penalties on those products which fail to follow standards. UN/DOT 38.3 is the standard which governs the transportation of dangerous goods, it includes a combination of significant stresses including environmental, mechanical, and electrical. These tests include Altitude Simulation, Thermal, Vibration, and Shock testing, External Short Circuiting, and Impact, Overcharge, and Forced Discharge testing. While these tests can be difficult to pass one after the other, they provide the end-product with the ability to be much more easily transported which will allow us a greater market reach due to ease of shipping. Failure to follow this standard means that the product must be transported as a Class 9 Hazardous Good which greatly limits transport options.

IEC 61960 and IEC 62133-2:2017 detail a lot of battery specifications as well, including the physical dimensions (which applies to manufacturers), along with electrical tests which range from charge and discharge performance to endurance in cycles and electrostatic discharge, tolerances, and markings/designations. Further, the latter is more specifically geared towards Safety and is

more in-line with our use, detailing the charge and discharge procedure, tolerances, terminal contacts, wiring, current, temperature, and voltage management, venting, and more while providing specification of testing procedures. Both provide a lot of information to battery manufacturers but they are still necessary in our case due to the possibility of misuse/mishandling of the batteries, these standards apply to secondary cells but may be helpful in guiding us in our primary system. IEEE 1725-2011 is another standard which details qualification, quality, and reliability of rechargeable batteries but is specifically geared to mobile phones and although our product is not a mobile phone it is a mobile computing device and will be interacting with the user's phone. This standard is useful due to it's more stringent requirements and supersedes 1625. IEC 62368-1 supersedes IEC 60065 and IEC 60950-1, and represents a shift in engineering principles, it is a standard for the safety of electrical and electronic equipment that classifies energy sources, prescribes safeguards, and gives guidance in regards to the applications of and requirements for those safeguards.

IEC 62133 further defines requirements and tests for secondary cells and batteries, specifically those which are sealed and portable. This standard mostly provides information on tests which should be run and a variety of requirements. This standard was introduced more recently as the demand for Li-ion will increase this decade due to widespread adoption of electric vehicles, it further ensures battery reliability and safety. It also details safety standards for secondary battery systems and thus applies to our Alkaline-based backup system. This standard tests for molded case stress, external short circuit, free fall, and overcharging of battery, it is much less stringent than UN 38.3. UL 2054 is one of the most intense standards on the list and includes about double the amount of tests as UN 38.3, comprising seven electrical tests, four mechanical tests, four battery enclosure tests, one fire exposure test, and two environmental tests. This standard is very challenging and should thus be our main point of reference, it does defer component cell level testing to UL 1642 though so that must also be considered. UL 1642 is a general safety standard for lithium batteries, rechargeable or not, and includes testing for short-circuiting, heating, temperature cycling, forced-discharge, and altitude simulation along with fire-exposure, flaming particles, projectiles, and explosions. This standard is further substituted by application specific ones.

Table 4.5.1.1: Battery Standards

| Standard | Description |
|---|---|
| UL 1642 | Standard For Lithium Batteries |
| UL 2054 | Standard For Safety of Household and Commercial Batteries |
| IEC 62133 | Standard for Exporting Lithium-Ion Batteries for IT, tools, Lab, Household, and Medical Equipment |
| IEC 60950-1 | Safety Standard for Battery-Powered IT Equipment |
| IEC 62368-1 | Newer Safety Standard for Battery-Powered IT Equipment |
| IEEE 1725-2011 | Standard for Rechargeable Batteries for Multi-Cell Mobile Computing Devices |
| IEC 61960 | Performance Standard for Secondary Alkaline cells and batteries for portable applications |
| IEC 62133-2:2017 | Safety Standard for Secondary Alkaline cells and batteries for portable applications |
| UN/DOT 38.3 | Recommendations on the Transport of Dangerous Goods |

### 4.5.2 Bluetooth

There are not many Bluetooth standards, the main one is IEEE 802.15.1. This standard defines the lower transport layers of the Bluetooth technology including L2CAP, LMP, Baseband, and radio. This standard goes into great detail and provides enough information for us to be able to utilize the technology, namely with regards to the physical layer and Media Access Control (MAC) specification. This technology utilizes a frequency in the radio band and is similar to Wi-Fi, IEEE 802.11, at 2.4GHz the frequency is the same but the protocol is what differs and really sets these two technologies apart.

Table 4.5.2.1: Bluetooth Standards

| Standard | Description |
|---|---|
| IEEE 802.15.1 | General Bluetooth Standard and Technology Information |

Table 4.5.2.2: Bluetooth Classes

| Type | Power | Range |
|---|---|---|
| Class 1 | High - 100mW (20dBm) | 91 meters / 300 feet |
| Class 2 | Medium - 2.5mW (4dBm) | 9 meters / 30 feet |
| Class 3 | Low - 1mW (0dBm) | 1 meter / 3 feet |

### 4.5.3 USB

USB does not have standards with from the most prominent standard-making bodies but instead has its own, the USB Implementers Forum or USB-IF for short. This body is responsible for maintaining all patents related to USB and is the "patent pool" for patents relating to USB. USB-IF licenses the patents under Reasonable and Non-Discriminatory terms without royalties. [43]

Table 4.5.3.1: USB Power Delivery by Type [44]

| Version | Max Power | Voltage | Max Current |
|---|---|---|---|
| USB 2.0 | 2.5W | 5V | 500mA |
| USB 3.1 | 4.5W | 5V | 900mA |
| USB BC 1.2 | 7.5W | 5V | 1.5A |
| USB Type-C 1.2 | 15W | 5V | 3A |
| USB PD | 100W | 5/9/15/20V | 5A |

### 4.5.4 Software Testing

The biggest standard for software testing is ISO/IEC/IEEE 29119 which is based on IEEE 829 and 1008 along with BS 7925-1 and -2. Before ISO WG26, there was no working group with significant software testing experience. WG26 intially developed four sections for this standard before publishing the fifth in 2016. The five parts are Concepts and Definitions, Tests Processes, Test Documentation, Test Techniques, and Keyword-driven Testing. Part one is an introduction, two provides a generic test process and higher level processes, three provides templates and documentation examples produced in testing including documentation at the organizational and management levels, four covers test design techniques and there are three main categories which are Specification-, Structure-, and Experience-based techniques, five covers keyword testing which

is more resilient to OS changes. [45] ISO/IEC 25010 relates to the use of SQuaRE which is the most recent, prominent series of international standards, this standard is most useful for identifying objectives and requirements. ISO/IEC 25000 on the other hand gives guidance on SQuaRE, including giving a general overview, reference models and definitions, and relationships between the documents.

Table 4.5.4.1: Software Testing Standards

| Standard | Description |
|---|---|
| ISO/IEC/IEEE 29119 | International Software Testing Standard |
| ISO/IEC 25010 | System and Software Quality Models |
| ISO/IEC 25000 | Systems and Software Quality Requirements and Evaluation - Guide to SQuaRE |

### 4.5.5 Language Standards

For our application we will be utilizing React-Native, it was created by Facebook as an open-source Javascript mobile app framework. Although there are no standards for React-Native there are different versions and inherently with coding there are different coding styles so it is the style which will be standardized below.

**Guidelines** [46]

- Follow the Component/Presentation Pattern
- Use Higher Order Components when possible
- Split code into respective files
- One React component per file
- Avoid inline CSS and follow CSS naming convention (BEM, SUIT)

Table 4.5.5.1: Coding Standards [46]

| Standard | Affected |
|---|---|
| PascalCase | Component Names |
| camelCase | Non-Component Names, Attribute Names, Inline Styles, Variable Names |
| Same Name as Corresponding File/Component | Unit Testing Files, CSS Files |

# 5 Project Hardware and Software Design Details

## 5.1 Initial Design Architectures and Related Diagrams

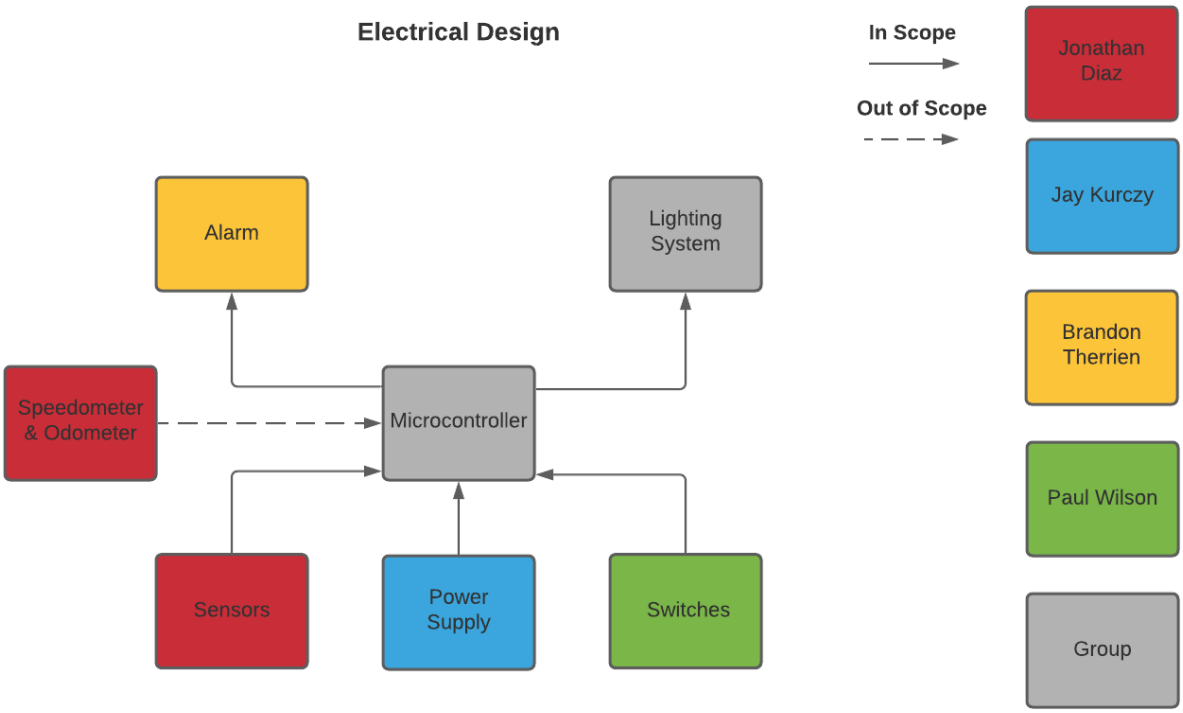



Figure 5.1.1: Electrical design block diagram

The figure above represents the electrical design of the project. The microcontroller is what the other systems will be communicating to, thus making it the heart of the project. The alarm is one of the outputs and the microcontroller will trigger it once the conditions are met. The microcontroller will also control the lighting system. The lighting system consists of the tail light, turn signals, and headlights. Now discussing the basics of the inputs to the microcontroller. Inputs involve sensors, which will be used to send signals to the microcontroller upon theft detection. The power supply is of course needed as an input for without it there simply would be no power provided to all of the systems. Switches are an additional input that will be used to tell the microcontroller whether the security system is armed as well as controlling the lighting system. The optional input to the microcontroller is the speedometer and odometer, which will be an additional defense for the anti-theft system.

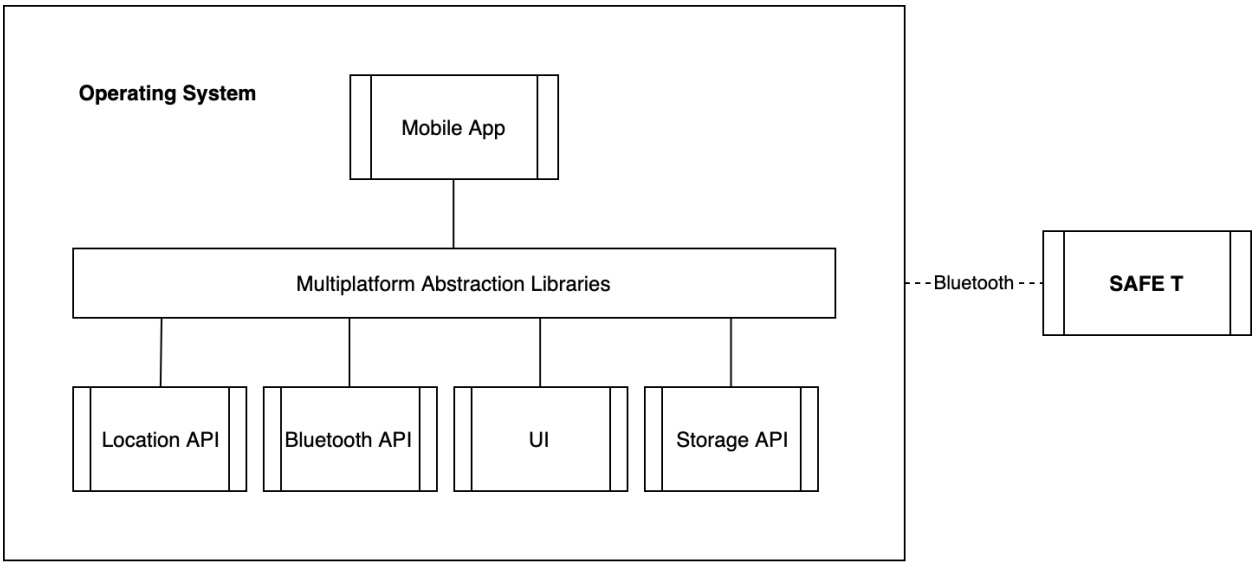



Figure 5.1.2: Software Design block diagram

The figure above is an abstracted look at the mobile application architecture we have planned. The application as a whole is a hosted software, meaning it runs entirely on the operating systems it is developed for and cannot function without, unlike the SAFE T module, which is an independent platform. We will be using a lot of the OS libraries (directly or indirectly), to gain access to a lot of information for the mobile application to enhance the utility of the product. While the APIs in the figure may not work the same exact way, there are various abstraction libraries that make it easier for developers to write applications utilizing those features, without compromising features on either operating system.

The basic control and data flow flow is depicted by the following diagram:

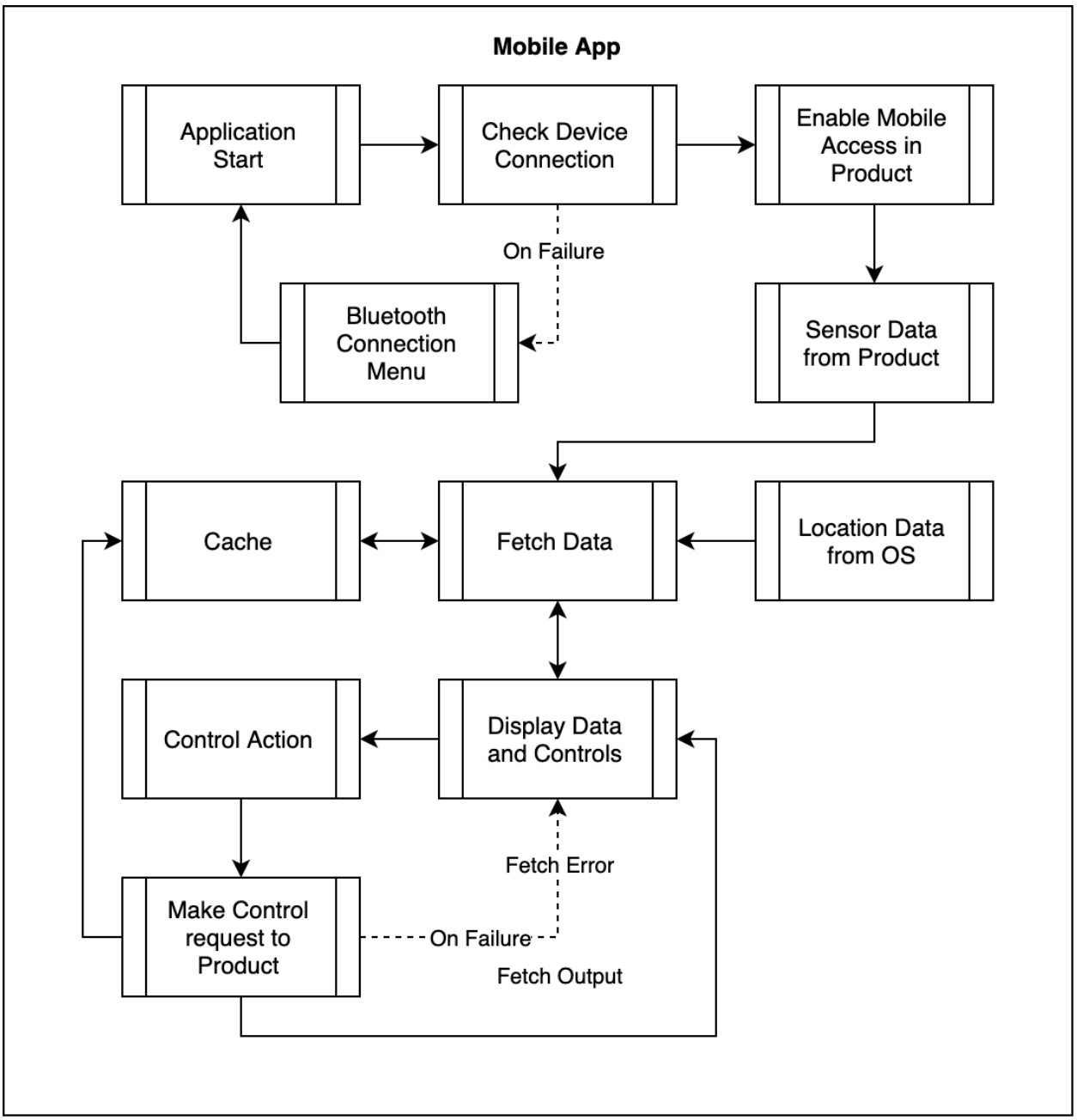



Figure 5.1.3: Application Data and Control Flow block diagram

# 6 Design Analysis

Most of the design testing/analysis can be done without measurements. That is, testing if the buttons work, the LEDs flash correctly, the siren is effectively loud and that the app works correctly can all be tested by the designers. Despite this, it is important to properly measure the rating of each output. The decibel rating of the siren can be measured either with an app or with professional sound equipment. The siren will be run at various distances and cross-checked with the calculations in order to see if (1) the output is satisfactory and (2) the output matches what should be expected. Similarly, the lumen output of the LED screens can be measured with a lux meter. The light should be tested at various lengths and at multiple daylight levels. These measurement tools will have to be factored into the budget.

The only thing that would require additional tools to measure would be the current through the wires, which can be accomplished with a common multimeter. This value can be cross-referenced with the specifications given by the battery and the microcontroller and by using common circuit analysis. Additionally, it may be possible to replicate the signal on circuit software such as NI MULTISIM.

For the software, including the programming of the microcontroller and the app, the same method of testing will be used; that is, the first test will be to see if all of the applications and signals function to their intended use with no adverse effects. Once that is done, the software can be tested using a variety of methods, refer to section 7.2.

## 6.1 Microcontroller Design

The microcontroller will be tasked with controlling and communicating with the head and tail light along with the turn signals for the lighting system, the vibration sensor and/or other sensors and the alarm for the anti-theft system, and the battery for power systems. The MCU's design must support our requirements and allow for the software described in Section 6.7. The design diagram is shown in Figure 6.1.1.

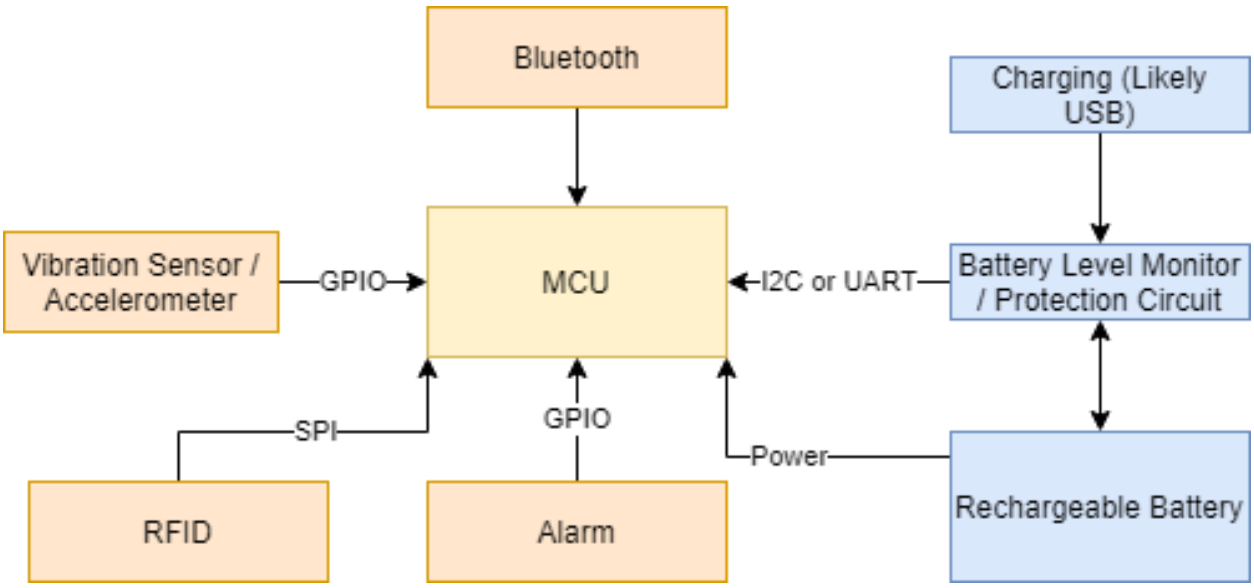



Figure 6.1.1: MCU Design Diagram

The lighting system will be implemented such that, for the head and taillights, the control signal will come from the MCU and be passed into a MOSFET Amplifier. The amplifier will provide the correct voltage for its respective LED. Each amplifier will be supplied power by the battery. By utilizing PWM, we may easily control voltage supplied through the MOSFET and thus the brightness of the lights. For the turn signals, we may similarly need an amplifier, but it would simply be turned on and off. For timing the signaling we will use the MCU's built-in interrupts, which will allow us to further utilize the onboard clock(s) in order to condense our circuitry and increase efficiency. By using interrupts, we may much more easily change and test the signal timing. Each interrupt will serve to toggle the respective turn signal.

The control system will take inputs from the sensors, namely the vibration sensor, and the buttons to control the head/tail lights. All inputs will be wired to appropriate pins on the MCU, mostly GPIO unless they require a different type. Depending on the MCU selected, the lights may need to be connected to a higher-powered pin.

The power system will send a battery charge level signal to the MCU for storage, as this is required for the mobile app. This signal will be decoded by the MCU OS and sent to the app for display. Depending on which MCU we select, the rechargeable battery will connect directly to it via USB or DC, this will also depend on the protection circuit and battery level monitor utilized.

### 6.1.1 PCB Design

To start the process of designing the group's own custom PCB microcontroller the group must first decide on what software/program will be utilized. There are a variety of options that can be found on the internet such as Altium Designer, KiCad, and Autodesk EAGLE. The key differences between the top two candidates for what the team will choose to design the microcontroller PCB is between EAGLE and KiCad. KiCad offers vaious SPICE simulation support and has an open source support making it free. One huge flaw with KiCad is that there is no auto routing feature

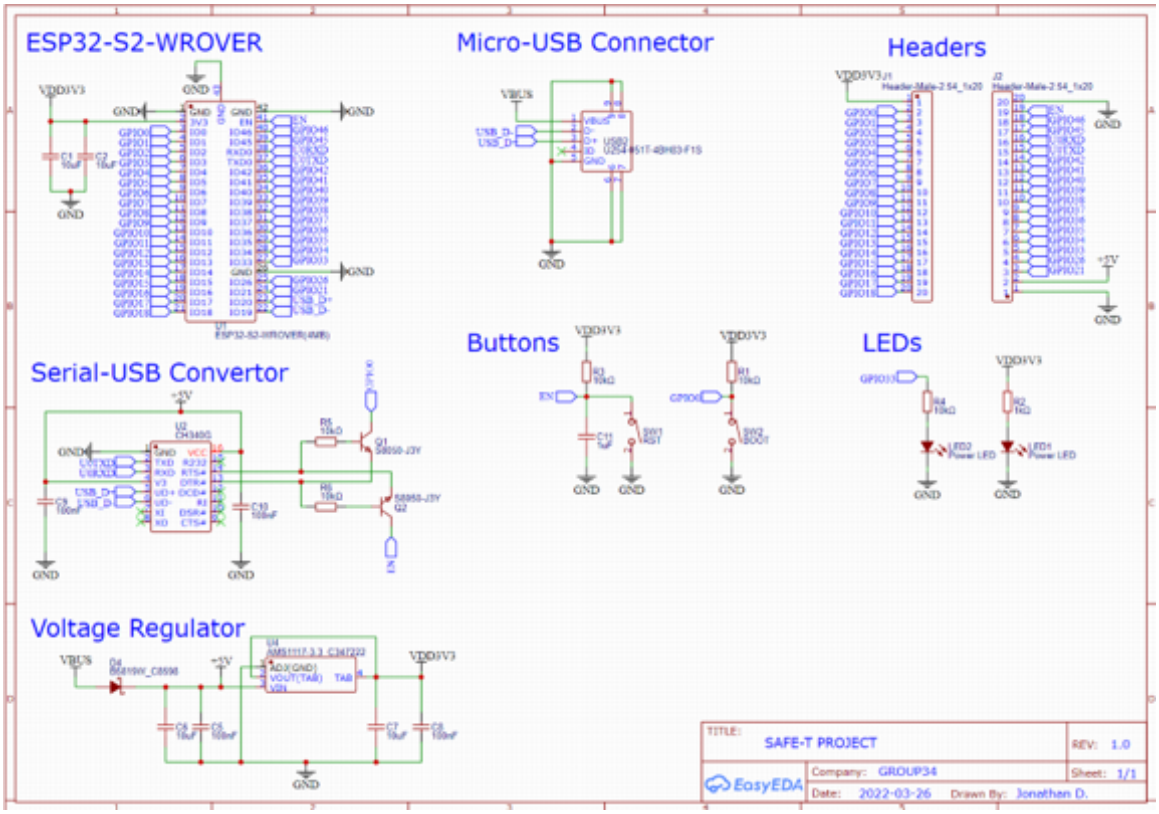
which makes routing all the signals for the PCB much more time consuming. With EAGLE, there is fantastic ERC and DRC check engine. ERC and DRC essentially covers the electrical and design rules so that the PCB design is ultimately error free. EAGLE also provides auto routing as well as simple BOM generation from selected components. A major flaw of EAGLE is that it requires the user to be a college student in order for the program to be free. Otherwise, the cost is very high.

The group has used Autodesk EAGLE the most from previous courses, particularly our junior design course. Thus, the group will decide on using Autodesk EAGLE, the student version, because it is free and the most familiar program out of the other options. Even if the group struggles with some designing concepts using Autodesk EAGLE, there are a multitude of tutorials that can be found on youtube or just on the internet in general because it is a popular program.

The process of the design then continues with creating a project and the schematic via Autodesk EAGLE. Since there are a variety of components that will be needed to create the project's custom microcontroller there must be a library of components to choose from, which is supplied by the base EAGLE library. If the supplied EAGLE library does not contain components needed for the design, then EAGLE allows importing other libraries that can be found on the internet. These libraries contain schematic symbols and PCB footprints that can make the designing process far less complicated.

The design of the custom PCB microcontroller can be split into major parts which include the microcontroller chip, a voltage regulator, headers, and a programming connector. Most components came from the provided EAGLE library as well as from SparkFun libraries. The full PCB schematic is shown in Figure 6.1.1.1. Each section of the PCB schematic will be discussed for better understanding of the design choices made.

Figure 6.1.1.1: MCU Design




### 6.1.1.1 Microcontroller Module

The microcontroller module features the ESP32-3C System on a Chip (SoC), which has a single-core MCU, as well as for 2.4G Wi–Fi and Bluetooth Low Energy version 5.0 combo. Its interface type features GPIO, I2C, I2S, SPI, UART which means it covers most communication protocols if needed. The program memory size is 4 MB and operating supply voltage is 3.3V. Overall the ESP32-3C SoC has everything the team needs for the project application. The ESP32-3C SoC holds various pins each holding different purposes. Some of these pins will be utilized while some may not.

To explain the design process, the ESP32-3C pins that will be used will be explained first. Starting with the VDD3P3_1, VDD3P3_2, VDDA_1, VDDA_2, and VDD3P3_RTC, VDDSPI, and VDD3P3_CPU pins, which all are connected to the system voltage regulator, power supply. Each of these pins require 3.3V to be supplied for any functionality, which is why all are directly connected to the voltage regulator output. Now that the input power pins are done next the CHIP_EN pin is connected such that when high (ON) the chip is enabled and if low (OFF) the chip powers off. It's also important to note that the chip enable pin does not allow floating. To avoid floating, a pull up resistor is implemented. The pull up resistor is a 10k resistor because the pin does not require much current. A switch is also needed so that once pressed a short will be introduced, which is why it's connected to ground in the schematic. This pin essentially will serve as a reset button for the microcontroller. Pins GPIO18 and GPIO19 are connected because both are USB pins. GPIO18 being USB_D- while the GPIO19 pin is USB_D+, both are directly

connected to a USB type C connector for debugging/programming purposes. The ESP32-C3 chip has a built-in JTAG circuitry which is why a USB-to-UART chip is not needed. The rest of the pins are connected to a header so when needed are accessible. A table is created to list all the pins the ESP32-C3 has, shown in Table 6.1.1.1.1.

Table 6.1.1.1.1: Functions of each ESP32-C3 pin from the data sheet [47]

| Pin Name | Pin No. | Function |
|---|---|---|
| LNA_IN | 1 | RF input and output |
| VDD3P3 | 2 | Analog power supply |
| VDD3P3 | 3 | Analog power supply |
| XTAL_32K_P | 4 | GPIO0, ADC1_CH0, XTAL_32K_P |
| XTAL_32K_N | 5 | GPIO1, ADC1_CH1, XTAL_32K_N |
| GPIO2 | 6 | GPIO2, ADC1_CH2, FSPIQ |
| CHIP_EN | 7 | High: on, enables the chip<br>Low: off, the chip powers off.<br>Note: Do not leave the CHIP_EN pin floating |
| GPIO3 | 8 | GPIO3, ADC1_CH3 |
| MTMS | 9 | GPIO4, ADC1_CH4, FSPIHD, MTMS |
| MTDI | 10 | GPIO5, ADC2_CH0, FSPIWP, MTDI |
| VDD3P3_RTC | 11 | Input power supply for RTC |
| MTCK | 12 | GPIO6, FSPICLK, MTCK |
| MTDO | 13 | GPIO7, FSPID, MTDO |
| GPIO8 | 14 | GPIO8 |
| GPIO9 | 15 | GPIO9 |
| GPIO10 | 16 | GPIO10, FSPICS0 |
| VDD3P3_CPU | 17 | Input power supply for CPU IO |
| VDD_SPI | 18 | GPIO11, output power supply for flash |
| SPIHD | 19 | GPIO12, SPIHD |
| SPIWP | 20 | GPIO13, SPIWP |
| SPICS0 | 21 | GPIO14, SPICS0 |
| SPICLK | 22 | GPIO15, SPICLK |

| Pin Name | Pin No. | Function |
|---|---|---|
| SPID | 23 | GPIO16, SPID |
| SPIQ | 24 | GPIO17, SPIQ |
| GPIO18 | 25 | GPIO18, USB_D |
| GPIO19 | 26 | GPIO19, USB_D+ |
| U0RXD | 27 | GPIO20, U0RXD |
| U0TXD | 28 | GPIO21, U0TXD |
| XTAL_N | 29 | External crystal output |
| XTAL_P | 30 | External crystal input |
| VDDA | 31 | Analog power supply |
| VDDA | 32 | Analog power supply |
| GND | 33 | Ground |

It's important to note that the ESP32-C3 has 22 General Purpose Input / Output Interface (GPIO) pins. All of which can be assigned various functions by configuring registers. All digital IO pins are bi-directional, non-inverting and tri-state based on the datasheet.

The schematic design of the microcontroller module is shown in Figure 6.1.1.1.1. The schematic shows every pin connection of the microcontroller for clear understanding.

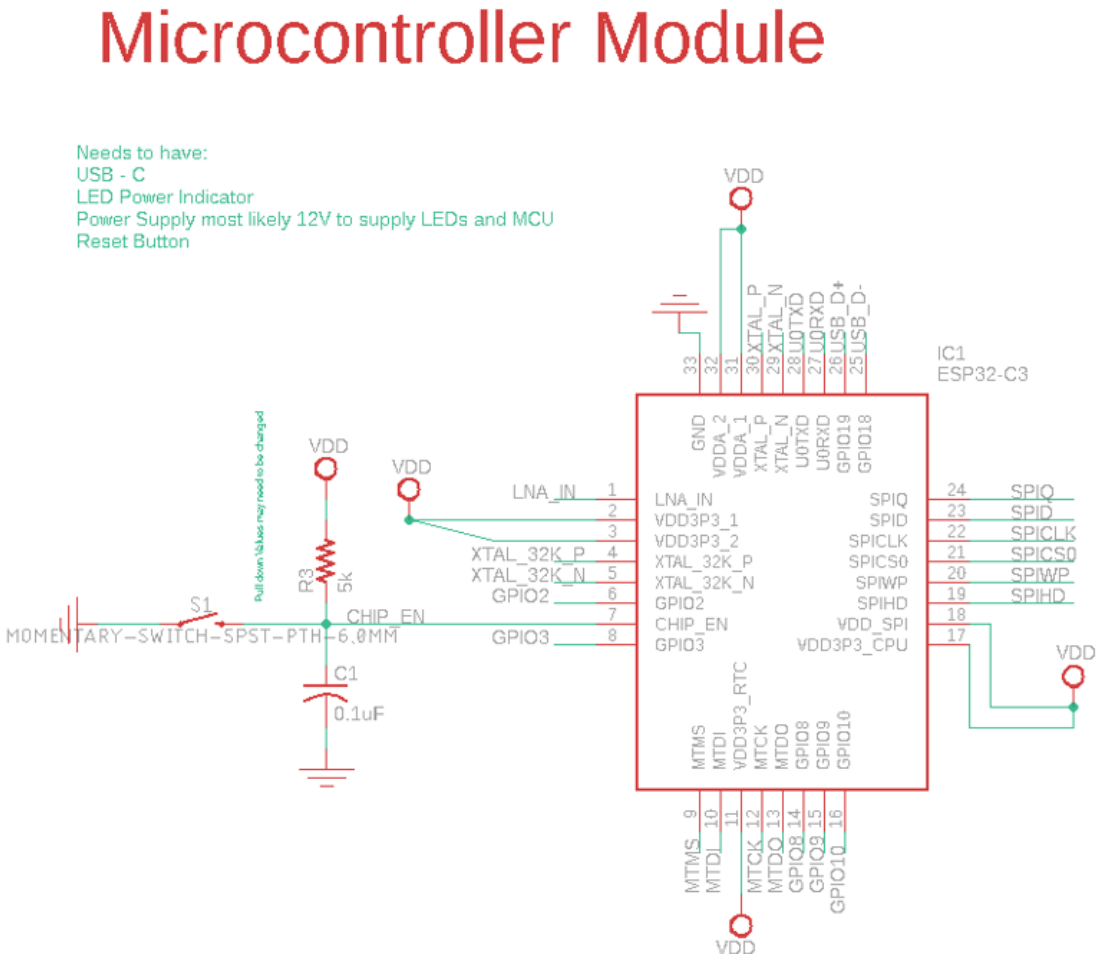



Figure 6.1.1.1.1: Schematic Design of the Microcontroller Module

Next the PCB design of the microcontroller module is shown in Figure 6.1.1.1.2. The PCB design for the microcontroller module mainly consisted of using various techniques. One of the techniques practiced was to preserve space, which translates to less copper being wasted. The team also made sure to have all peripheral connections placed at the edges of the board for easy accessibility. Lastly, the PCB design required all components to avoid any route intersections. For example, the microcontroller was placed at the center of the board due to the fact that it has the most connections.

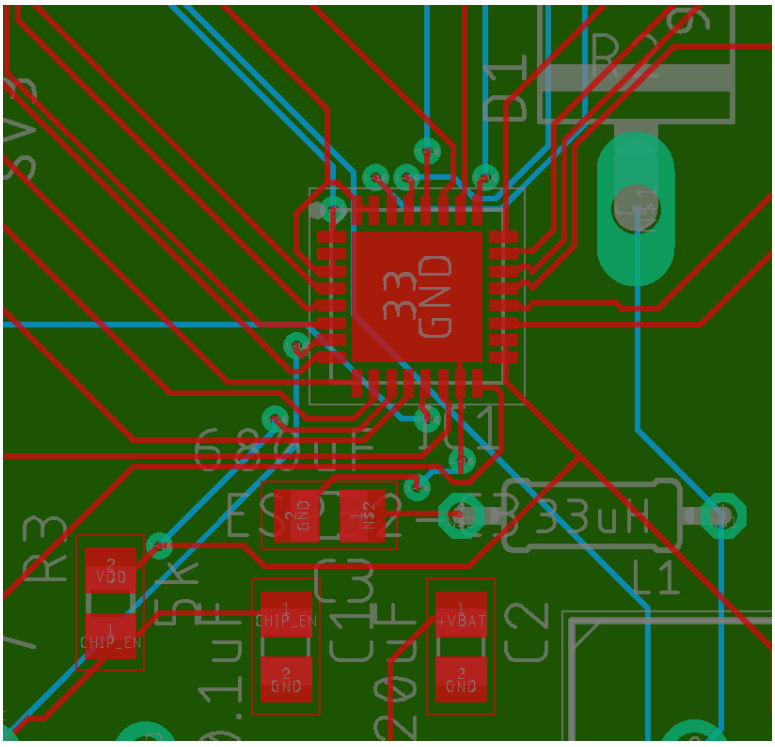



Figure 6.1.1.1.2: PCB Design of the Microcontroller Module

### 6.1.1.2 Debugger/Programming Connector

A debugger and programming connector is needed to be able to debug and program the ESP32-C3. The ESP32-C3 has built in JTAG circuitry to allow a USB cable to connect to the D+ and D- pins. Thus, the team decided to connect the D+ and D- pins from the microcontroller directly to a USB type C female connector. This USB type C connector will simply be connected to the D+, D-, external voltage supply, and ground. The idea of this USB type C port is that a USB cable can directly connect to the ESP32-C3 to debug and program it, which is essential. Shown below is the schematic of the microcontroller, Figure 6.1.1.2.1. The USB type C with needed pins are connected, while others that will not be useful for the design remain unconnected.

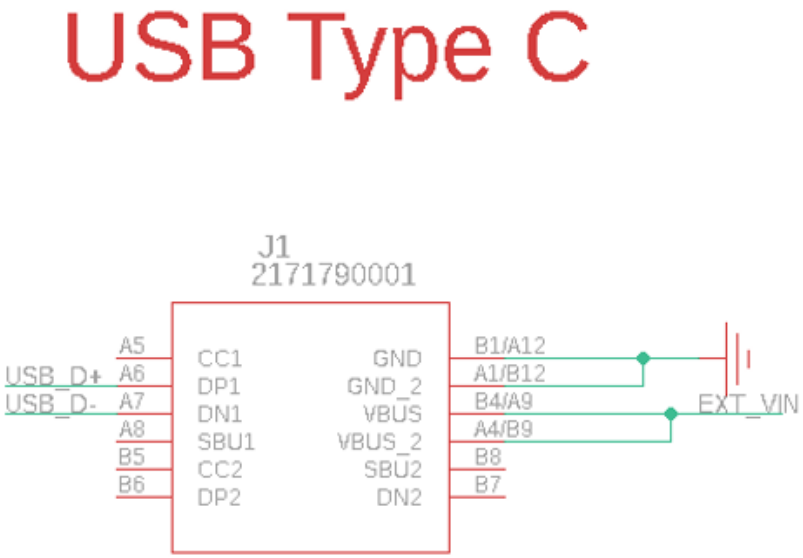



Figure 6.1.1.2.1: Schematic of the USB Module

The USB type C PCB layout is shown in Figure 6.1.1.2.2. The PCB design of the USB type C was done so that its close to the edge of the PCB board for easy accessible connection.

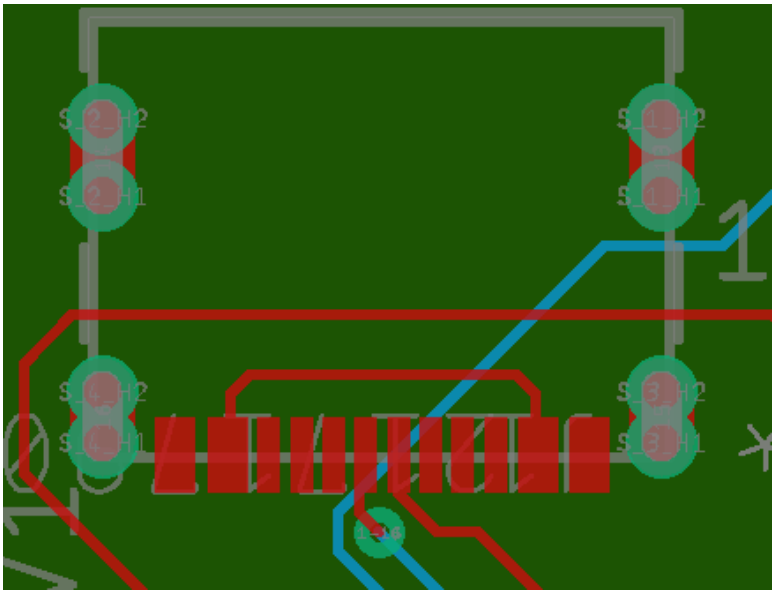


Figure 6.1.1.2.2: PCB Design of the USB type C Module

### 6.1.1.3 Power Supply

A power supply is needed for without it the microcontroller simply would not operate at all. A voltage regulator is required as well for the microcontroller PCB design due to the required voltage

of the microcontroller limited to 3.3V. The step down voltage regulator was created through Texas Instruments power designer tool WEBENCH. WEBENCH offers customizable power supply circuits based on requirements. The design requirements of the voltage regulator was simple in that the overall circuit needed to convert 12V input to output 3.3V. With other considerations like cost and efficiency, the group wanted a balanced candidate which WEBENCH offers. The power supply circuit consists of the LM2596, a buck switching regulator capable of driving a 3A load and available fixed output voltages of 3.3V, 5V, and other voltage outputs up to 37V. The LM2596 is also capable of handling an input voltage of up to 40V, which is more than enough for our application. With the buck converter selected WEBENCH automatically picks the correct resistor, inductor, diode, and capacitor values to complete the circuit. The circuit with all electronic components is designed to convert 12V DC to 3.3V, which is what is needed to power the microcontroller.

Once the circuit was selected from WEBENCH the tool allows the circuit to be exported as a SVG file which can then be imported into EAGLE. In EAGLE the power supply was further modified to incorporate an LED that will indicate if the circuit is ON/OFF. The schematic of the power supply for the microcontroller design is shown in Figure 6.1.1.3.1

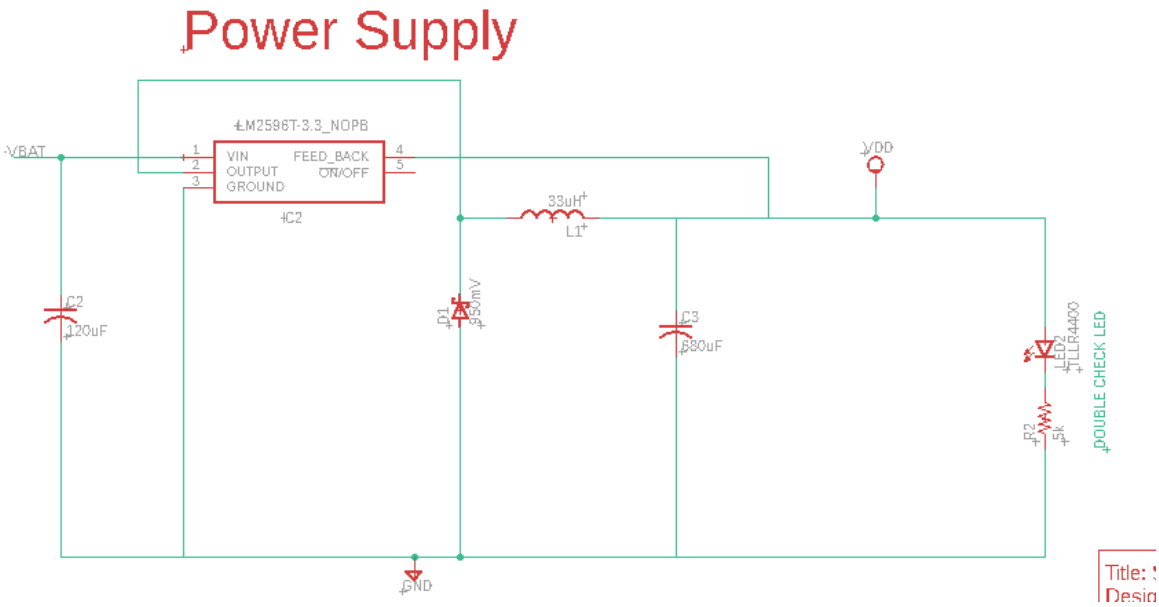



Figure 6.1.1.3.1: Schematic of the power supply

Next, the PCB layout of the power supply is shown in 6.1.1.3.2. The power supply was placed at the bottom of the PCB layout along with the 2 pin header simply because it serves as an easy connection to the board.

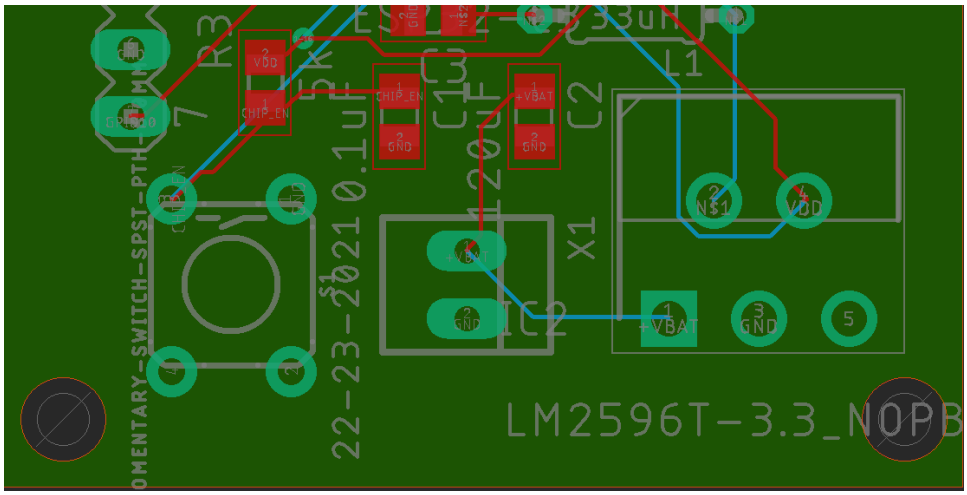



Figure 6.1.1.3.2: PCB layout of the power supply

### 6.1.1.4 Headers

The last portion of the schematic design involves the headers. Headers are essential because they provide accessibility to the ESP32-3C pins. Without them direct connections would be tedious. There are 4 pin headers: 1x4 pins, 1x5 pins, 1x6 pins, and 1x7 pins. The 1x4 pins consist of the XTAL_P, XTAL_N, U0TXD, U0RXD, 1x5 pins include the LNA_IN, XTAL_32K_P, XTAL_32K_N, GPIO2, GPIO3 signals, 1x6 pins has the SPI connections, and lastly the 1x7 pins hold the MTMS, MTDI, MTCK, MTDIO, and the rest of the GPIO signals which are GPIO 8 to 10.

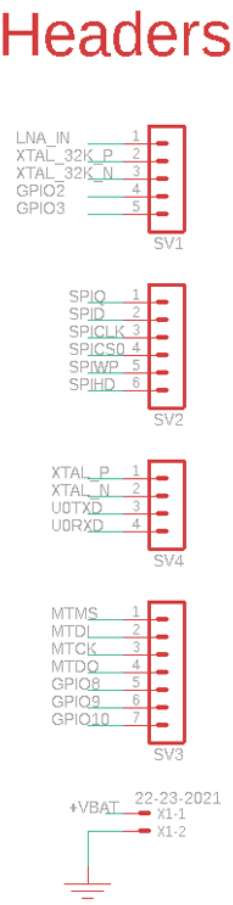



Figure 6.1.1.4.1: Schematic of the headers

Headers will be on the PCB design to reduce any size issues since any component that requires the needed microcontroller pins can be directly connected to the header rather than the use soldering. Headers also helps the design be far simpler rather than designing other components like the sensor to be integrated with the microcontroller PCB design. Below is the schematic of the headers, shown in Figure 6.1.1.4.1

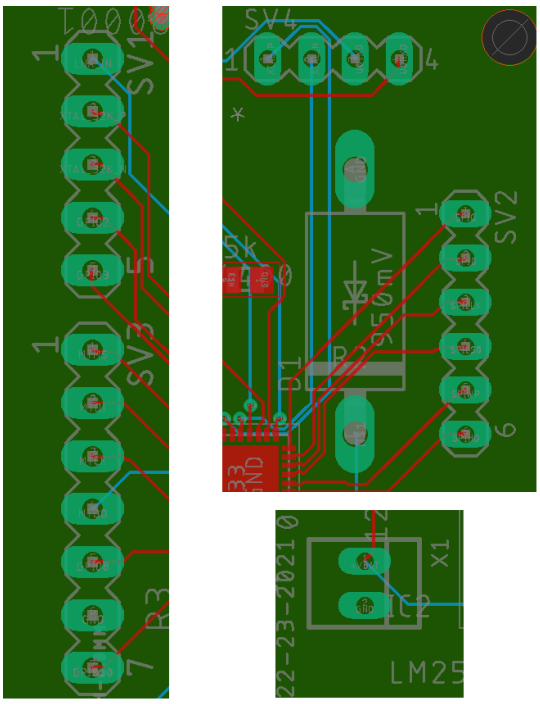



Figure 6.1.1.4.2: PCB of the headers

The overall microcontroller PCB design is shown below 6.1.1.4.3

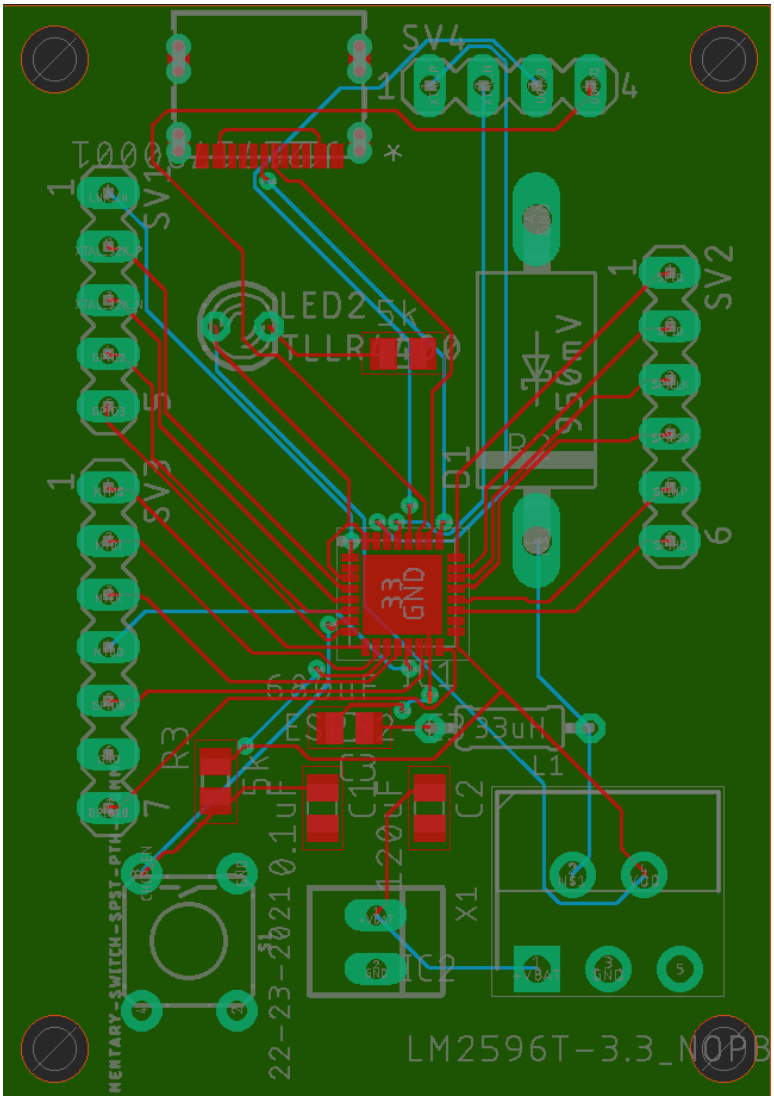



Figure 6.1.1.4.3: PCB of the the whole microcontroller

### 6.1.1.5 Bill of Materials

A list of all the components required for the PCB microcontroller design is mainly created to better understand the cost of each component. All components were selected from various models and manufacturers based on specific needs of the PCB design. The components ranged from simple electronic components like resistors and capacitors to voltage regulators as well as for the actual microcontroller SoC. The selection of each component besides stemming from the required values of the circuit design like a 5k resistor came with the need of spacing as well as component cost effectiveness. With that in mind, once all components for the circuit schematic was done through EAGLE, the program generated the bill of material. The generated bill of material for our PCB design was then exported as a text file which was then imported to Microsoft Excel. Editing was made on Microsoft Excel to delete certain criteria related to the bill of materials such as the order code and number of availability in regards to each component. These sections were deleted because they were not necessary since the team only selected components that are currently in

stock as well as for order code not giving any importance. Once deleted, some components like the switch used in the design as a reset button, the MCU SoC, and voltage regulator did not have all information in terms of price and price. This was due to EAGLE using only Newark for selecting components. To populate the missing information the team used other alternatives to purchase the components by using Digikey and Mouser. Finding the exact model number on either of the websites helped populate the missing information. Once all information was collected for each component the bill of materials was finished.

The bill of materials consists of dividing each component based on its quantity, value, package, manufacturer, manufacturer code, a brief description, and price. All of this is shown below in Table 6.1.1.5.1. It's crucial to list all of this in the bill of materials in case a certain component fails and needs to be replaced or troubleshooted through the manufacturers provided datasheet.

Table 6.1.1.5.1: Bill of Materials of the PCB Microcontroller

| Qty | Value | Package | Manufacturer | MPN | Description | Price |
|---|---|---|---|---|---|---|
| 1 | 0.1uF | C0805 | TDK | C0603X5R1C104K030BC | SMD Multilayer Ceramic Capacitor | $0.023 |
| 1 | 120uF | CAPSMT_62_F61 | PANASONIC | EEU-FC1V121 | Electrolytic Capacitor | $0.126 |
| 1 | 680uF | SM_RADIAL_H13 | PANASONIC | EEU-FP1E681 | Electrolytic Capacitor | $0.36 |
| 1 | 950mV | DPAK | INFINEON | BAS12504WH6327XTSA1 | Small Signal Schottky Diode | $0.232 |
| 1 | ESP32-C3 | QFN50P500X500X90-33N | Espressif Systems | ESP32-C3 | ESP32-3C SoC | $1.13 |
| 1 | 2171790001 | 2171790001_MOL | MOLEX | 217179-0001 | USB Type C | $0.473 |
| 1 | 33uH | MSS1210 | BOURNS | CM453232-330KL | Inductor | $0.16 |
| 1 | TLLR4400 | LED3MM | VISHAY | TLLR4400 | Red LED | $0.261 |
| 2 | 5k | R0805 | VISHAY | CRCW08055K00FKTA | SMD Resistor | $0.225 |
| 1 | | MA05-1 | MOLEX | 22-28-4051 | Pin Header | $0.075 |

| Qty | Value | Package | Manufacturer | MPN | Description | Price |
|---|---|---|---|---|---|---|
| 1 | | MA06-1 | MULTICOMP PRO | 2211S-06G | Pin Header | $0.049 |
| 1 | | MA07-1 | MOLEX | 90120-0127 | Pin Header | $0.217 |
| 1 | | MA04-1 | MOLEX | 22-28-8042 | Pin Header | $0.254 |
| 1 | MOMENTARY-SWITCH-SPST-PTH-6.0MM | MOMENTARY-SWITCH-SPST-PTH-6.0MM | SparkFun | COM-09190 | Push Button | $0.50 |
| 1 | LM2596T-3.3/NOPB | TO-220-5 | Texas Instruments | LM2596T-3.3/NOPB | Voltage Regulator | $6.12 |
| 1 | 22-23-2021 | 22-23-2021 | MOLEX | 22-23-2021 | Pin Header | $0.10 |

## 6.2 Anti Theft Alarm Design

The alarm will be designed such that if the security sensor detects vibration over a threshold point the alarm will turn on. Other conditions will need to be met in order for the sensor to be turned on. These conditions include whether or not the alarm system is armed via phone app or physical button and whether the Radio Frequency Identification is near proximity of the bike. With all conditions met, the sensor is armed and will determine thievery through vibration detection which will trigger the alarm if the threshold is exceeded. To further illustrate how the alarm security system operates Figure 6.2.1 is shown below.

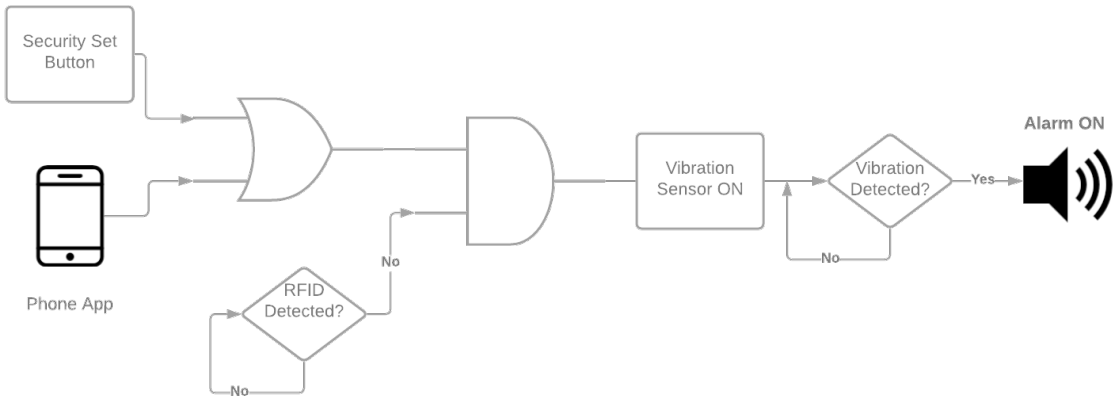



Figure 6.2.1: Alarm Security System for Alarm to Turn ON

In the condition the alarm turns on the owner of the bicycle needs to either use the phone app or RFID to reset the alarm. Either of the two conditions will turn off the alarm which is shown in Figure 6.2.2.

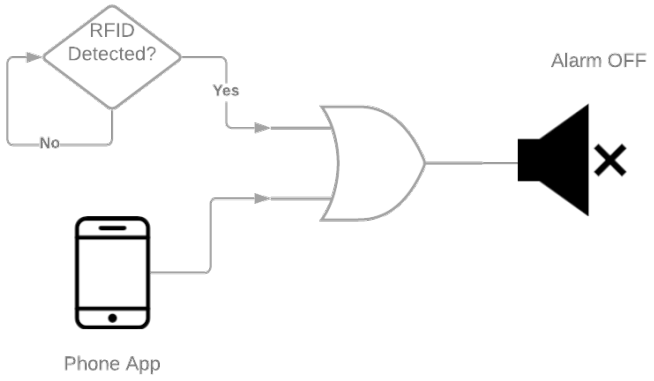



Figure 6.2.2: Alarm Security System for Alarm to Turn OFF

With the high level block diagram design done the team can move on to the parts that will be needed for the system design. The parts required include a RFID, an alarm, and vibration sensor. The sensor will have its own section to be discussed.

### 6.2.1 RFID Specifications

An RFID is required because it will serve as a key to lock and unlock the security system. Essentially it acts like a car keyfob, its tag can communicate with the RFID reader via radio waves. Once the RFID tag and reader are close enough the communication is established it becomes activated. The necessary RFID will have a small frequency range so that the owner of the bicycle needs to be very close to access control of the alarm system. Specifically, the RFID range needs to be regarded as low frequency, which is 30 - 300 kHz [48]. The read range needs to be up to 10 centimeters. Another important requirement for choosing the correct RFID reader is that the RFID reader needs to be compatible with the microcontroller the team will design. The RFID reader that will be utilized that meets these requirements is the TMS37157. The cost of the TMS37157 is $1.836 from Texas Instruments website. The specifications are listed in Table 6.2.1.1.

Table 6.2.1.1: TMS37157 Specifications

| Model | Frequency | Interface | Voltage - Supply |
|---|---|---|---|
| TMS37157 | 134.2kHz | SPI | 2V - 3.6V |

The TMS37157[49] features ultra low power consumption, 3 wire SPI interface, and 121 bytes memory. It can operate without use of a battery and will utilize low frequency to establish communication with the RFID tag.

The RFID tag that will communicate with the RFID reader will be the TRPGR30ATGA[50]. This specific model is a Texas Instruments passive RFID with a frequency of 134.2kHz.. This means it should be compatible with the RFID reader, TMS37157. A diagram of its dimensions is shown in Figure 6.2.1.1. It was found on the Digi-key website and costs $6.05.

Figure 6.2.1.1: Size Dimensions of the TMS37157

On the datasheet[51], it specifies that the EMC cannot be affected by natural electromagnetic interference or x-rays which is a bonus since some RFID tags are easily susceptible to electromagnetic fields. In regards to weight it weighs only 0.8 grams, so it's very light weight and easy to carry around. The signal penetration is stated to be read through almost all non metallic material.

### 6.2.2 Speaker Specifications

Speakers will be required for the security alarm system in such that once the vibration sensor detects enough vibration on the bicycle as well as no RFID presence the microcontroller will send a signal to activate the speaker to turn on. The speaker options mentioned in section 3.2.2 will be discussed again except this time the specifications of the speakers will be told in more detail to help with the design process of the alarm system.

Table 6.2.2.1: Speaker Size Dimensions

| Speaker | Mono Enclosed Speaker with Plain Wires (Figure 3.2.2.1) | Speaker - 0.5W (8 Ohm) (Figure 3.2.2.2) |
|---|---|---|
| Length | 70 mm | 50mm diameter |
| Width | 31 mm | |
| Depth | 17 mm | 16 mm |
| Weight | 25 g | N/A ($<$ 1 lb) |
| Cable Length | 30 cm | None |
| Impedance | 4 Ω | 8 Ω |
| Power | 3 W | 0.5 W |
| Price | $3.95 | $2.25 |

The speaker chosen out of the two choices will be the 0.5W speaker because the size, cost, power consumption, and impedance intake is better than the Mono speaker. The 0.5W speaker has higher impedance than the Mono model meaning it isn't as susceptible to blow outs. The power required for the 0.5W speaker is also way less demanding which is beneficial in regards to preserving the battery life of the entire project design. The size dimensions are shown below in Figure 6.2.2.1 [11], supplied by the manufacturers website on SparkFun.

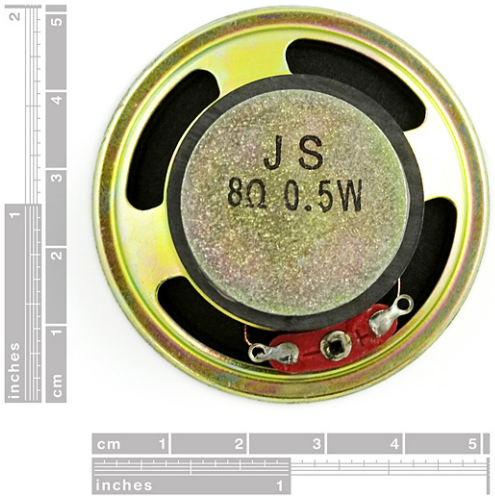



Figure 6.2.2.1: Speaker - 0.5 W (8 Ω) 50 mm diameter, 16 mm high, 28 mm base diameter

The speaker will be installed on the microcontroller, thus enclosed with proper housing to provide waterproof and dustproof security. The housing will also be designed in such a way that the sound

output from the speaker won't be quiet and distorted. This can be accomplished by designing the enclosure to help push out the volume better.

## 6.3 Battery Design

As mentioned before, the battery is the lifeforce behind our product and must thus be seriously considered and carefully designed. The top selections for this product are NiMH - Nickel Metal Hydride and Li-ion - Lithium Ion. This is due to a variety of factors which were discussed in Section 3.3. Both of these battery types will require a protection circuit. The power system is diagrammed in Figure 6.3.1. The Power Delivery Subsystem will vary depending on the final design but will comprise necessary voltage and current dividers or step-up circuits.

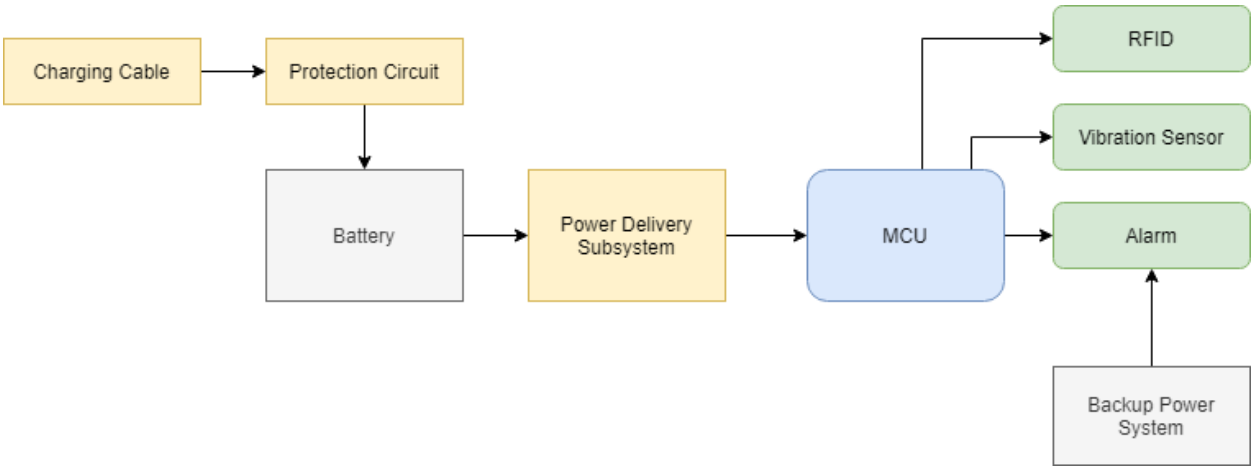



Figure 6.3.1: Power System Diagram

Due to the addition of a protection circuit, there are two options in this portion of the design. We may either create a protection circuit for each battery and reconfigure the product as further necessary or we may instead purchase batteries with the circuit built in. Creating our own protection circuits allows us to save money and have control over the components used but it disregards our limited knowledge versus that of the manufacturer, further increases workload, and heightens the chance of failure by decreasing product safety.

Battery voltage is not a major worry as a voltage divider circuit may very easily and cheaply be implemented. The only issue arises in creating a power bank capable of outputting a voltage high enough to support the lighting and alarm. The alarm max voltage should be less than four volts, utilizing voltage equal to the square root of power times resistance. The main concern is then the lighting voltage, which, according to Section 6.4.1.2, will be 12 - 13V. This means that our battery voltage should ideally output approximately 12.5V. This requirement puts us in a tough spot as our only options are to either put multiple low-voltage batteries in series or to buy a larger battery which operates at the required voltage. The former is preferred as we may more easily arrange multiple smaller batteries but the bulk would come from the charger whereas the latter would likely come with the charge circuit built in. Comparisons are detailed in Table 6.3.1.

Table 6.3.1: Battery Comparisons

| Column | A [52] | B [53] | C [54] | D [55] |
|---|---|---|---|---|
| Battery | 12V Lithium Ion w/ Built-in BMS | AmazonBasics AA NiMH | Samsung 25R 18650 | Panasonic NCR 18650B |
| Price | $54.99 | $34.99 | $6.99 ea * 4 = ~$28 | $9.99 ea * 4 = ~$40 |
| Amount | 1 | 24 | 4 | 4 |
| Capacity | 10 Ah | 2400 mAh | 2500 mAh | 3400 mAh |
| Voltage | 12.8 V | 1.2 V | 3.6 V | 3.6 V |
| Length | 6 inches | 1.99 inches | 2.56 inches | 2.73 inches |
| Width | 3.7 inches | 0.57 inch diameter | .71 inch diameter | .744 inch diameter |
| Depth | 2.6 inches | | | |
| Weight | 2.25 lbs | 1.52 lbs | 0.4 lbs | 0.42 lbs |
| Requires Addt. Circuit | No | Yes | Yes | No |
| Design | N/A | 10 in series 2 parallel sets Figure 6.3.2 | 4 in series | 4 in series |
| Net Capacity | 10Ah | 4800mAh | 2500mAh | 3400mAh |
| Net Voltage | 12.8V | 12.0V | 14.4V | 14.4V |

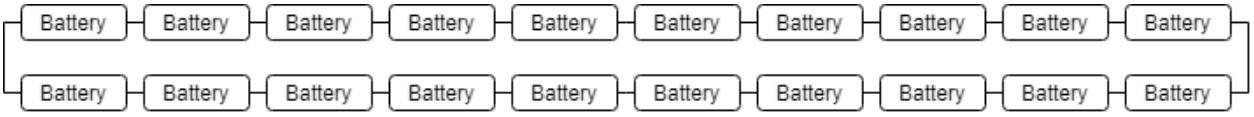



Figure 6.3.2: Table 6.3.1 Column B Battery Layout

For this product the ideal choice is the one which has the lowest weight and cost, smallest size, highest capacity, and does not require additional circuitry. That makes Table 6.3.1 column D the best choice, not only are the cost, weight, and size kept to a minimum the net voltage and capacity are well above the minimum required. The Panasonic NCR 1850B is shown below in 6.3.3.

Figure 6.3.3: The Panasonic NCR18650B 3400mAh 4.9A

Power through the MCU will be routed to the Alarm, Signals, and Lights. This is diagrammed in Figure 6.3.4 below. It is here where the signals to the head and tail lights are amplified through a MOSFET. The Left and Right signal will receive power through the MCU as well and will simply be toggled on and off.

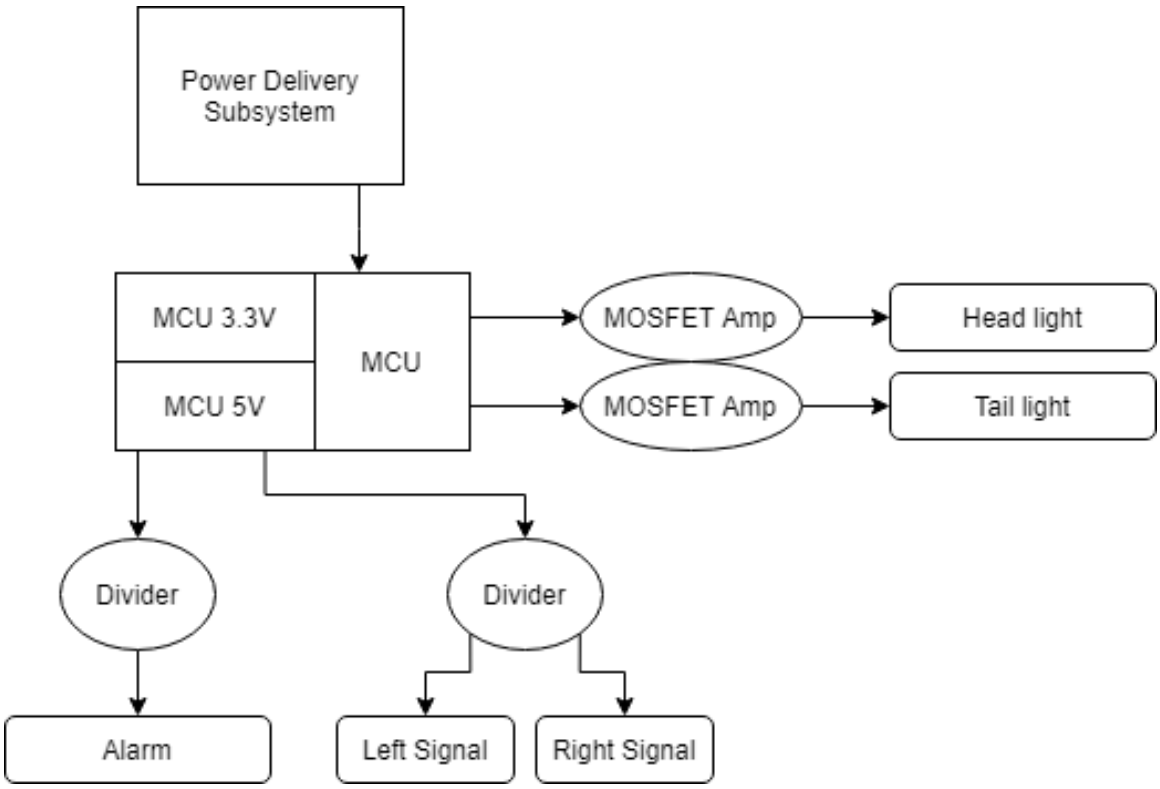



Figure 6.3.4: MCU Power System Layout

### 6.3.1 Backup Battery

This system must be made such that once power from the main battery is no longer detected, the backup immediately begins to power. The purpose of the backup battery is to power the alarm in case of theft attempts, this will require the MCU to be powered or to be bypassed which further requires the vibration sensor to be powered. By using a 9V battery, we may utilize a very common and cheap design that maintains a multi-year shelf life. This setup will require a simple design to detect main power, by using an active high signal and inverting, as demonstrated in Figure 6.3.1.1. Then the signal will be sent through an AND gate with the 9V battery, the gate will then connect to a voltage divider and further circuitry labeled Power Delivery Subsystem or PDS.

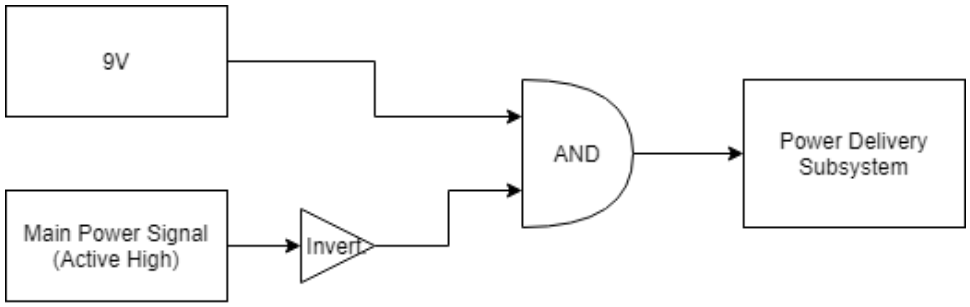



Figure 6.3.1.1: Backup Power System

## 6.4 Lighting Design

### 6.4.1 Headlight

The headlights will be designed in such a way that it will be the brightest compared to the rest of the lighting system. This is required due to the front view of the bicyclist being crucial especially during nighttime. The recommended amount of lumen required for the cyclist depends on the setting such that if the cyclist plans on riding in a lit urban area then 50 - 200 lumens will suffice meanwhile areas that don't have lighting will require at least 400 - 600 lumens. It is also imperative that the headlight is waterproof and dustproof, hence it must be IPX5. The headlights will also be controlled through a controller on the handlebar of the bicycle. The controller will be able to adjust the brightness of the headlights.

To approach the design of the headlights the selection of the type of LED technology that will be used was discussed in section 3.4. The team has decided to go through the selection of the COB LEDs due to the brightness intensity, cost, and low power consumption it offers. The color of the COB LEDs for the headlights will only be white for clear visibility. In terms of efficiency of the lighting projection the headlights will need to include a light reflector on top to prevent light scattering. It's also important to note that because of the power consumption the LED will need a heatsink so the LED doesn't overheat after a certain period of time.

The design overall will include: 10-12V 10W COB LEDs, 100x100x18 mm aluminum cooling heatsink, lens, and reflector.

#### 6.4.1.1 COB LEDs Specifications

The COB LEDs rated power is 20W with Forward Voltage (VF) 12-13V DC while the Forward Current (IF) is 1800mA. We wanted a powerful yet low power consumption for the LED so that it's less taxing in terms of the power consumption. The LED chip beam angle is 135-140 degrees and will produce 1400-1700 lumen. The size dimensions are shown below in Figure 6.4.1.1.1.

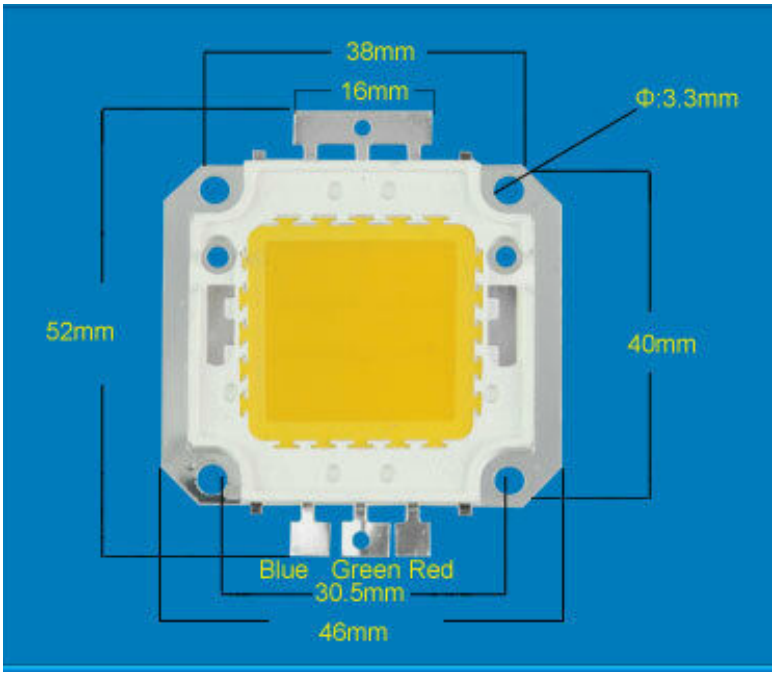



Figure 6.4.1.1.1: 20W 12V COB LED Size Dimensions [56]

The team potentially could've used other variations of the COB LED chip, however as mentioned before, the power consumption would've been too great for the system to handle or the brightness would be too insufficient. However, it is still important to compare the variations to demonstrate the differences, shown in Table 6.4.1.1.1.

Table 6.4.1.1.1: COB LED Chip Differences [57]

| Type | Quantity | Voltage | Lumen | Current | Price |
|---|---|---|---|---|---|
| 10W | 9 LEDs | 12-13 V | 900-1000 lm | 900 mA | $0.99 |
| 20W | 20 LEDs | 12-13 V | 1400-1700 lm | 1800 mA | $2.27 |
| 30W | 30 LEDs | 12-13 V | 2100-2550 lm | 2500 mA | $2.85 |
| 50W | 50 LEDs | 12-13 V | 4000-4500 lm | 4200 mA | $4.10 |

The table shows that once the COB LED chip is 30W+ the necessary current jumps up to 2500mA. Although the lumen produced is significantly greater than the 20W counterpart, the current is far too great and would create an obstacle in regards to the heat dissipation it would create. Thus, the 20W COB LED chip is chosen and will suffice as the headlights for the lighting system.

All information from this section was pulled from the manufacturer's website and is cited in the references for information traceability.

### 6.4.1.2 Aluminum Cooling Heatsink Specifications for Headlight

It is vital to utilize some form of heatsink for an LED especially the higher the wattage it consumes. If the temperature is unchecked the LED performance will deteriorate as time goes on which will affect its lumen output as well as overall lifetime. A depiction of what happens as the temperature increases is shown in Figure 6.4.1.2.1.

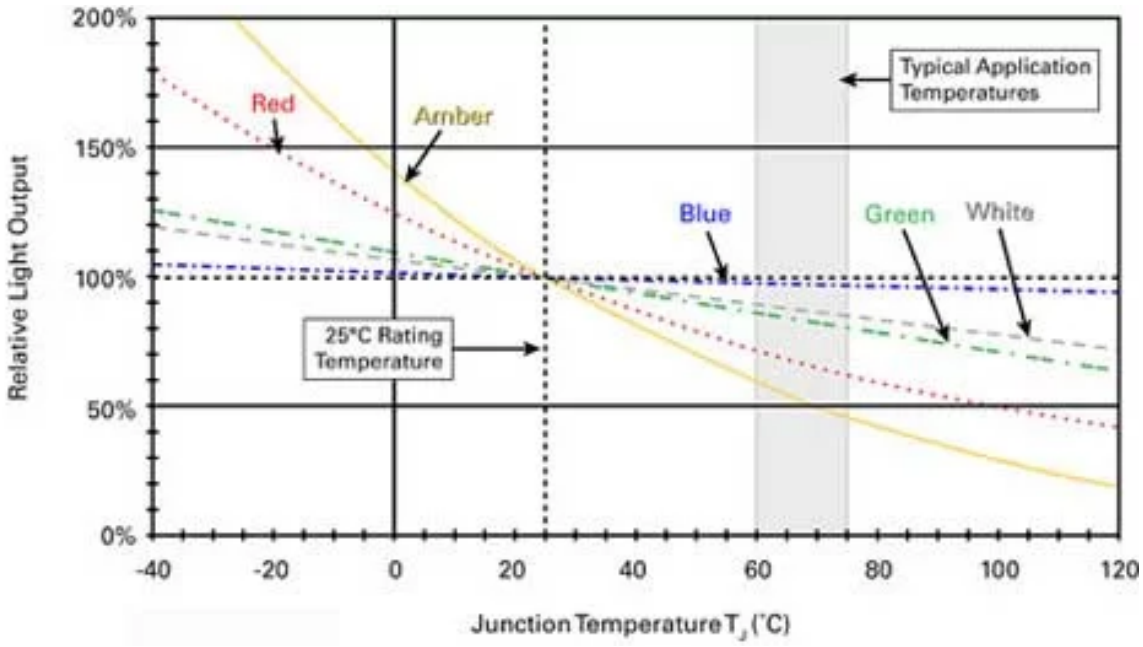



Figure 6.4.1.2.1: Junction Temperature of an LED vs Light Output[58]

The graph also demonstrates that based on the color of the LED the temperature can drastically impact its lumen output as the temperature increases. Since the headlights will be a white LED the performance can drop from 100% to 80% as temperature increases. To avoid severe temperature issues an aluminum heatsink will be used to attach to the COB LED. Aluminum alloy is a great choice in terms of the type of material for the heatsink because it can be shaped easily and is lightweight. The cost of an aluminum heatsink is also relatively cheaper than that of copper ones, further making it a solid choice. The lightweight feature of using aluminum can help significantly because the design of the headlights needs to avoid being bulky and heavy so that the chance of it dropping during a bicycle ride is low. Also, it will help the clip that will be holding the headlights in place be easier to design since weight won't be much of an issue. A depiction of the Aluminum Radiator that will be used is shown in Figure 6.4.1.2.2.

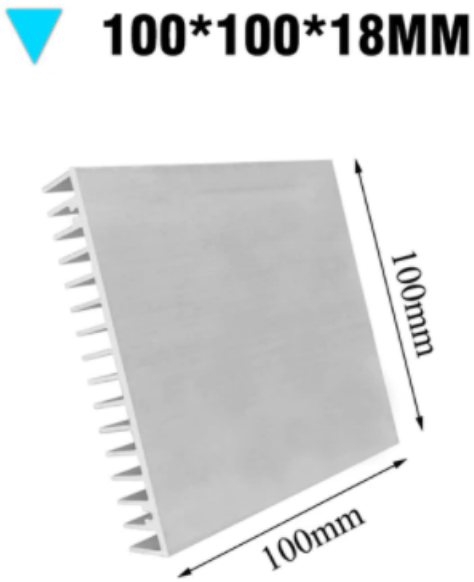



Figure 6.4.1.2.2: 100 mm x 100 mm x 18 mm Aluminum Heatsink [59]

### 6.4.1.3 Lens and Reflector Collimator Specifications for Headlight

A LED lens and reflector collimator will be required for the design of the headlights because the combination of these materials will reduce the scattering of the light beam. Preventing scattering of the light beam for the headlights will help concentrate the light output forward from the viewer rather than splitting and having a portion of light be shown on top of the LED. The team has selected a 44mm LED lens light reflector collimator that supports 60-80 degrees of the light beam angle. The material used for the lens is optical glass and is compatible with 20W LEDs. The size dimensions of the lens are shown in Table 6.4.1.3.1 as well as the dimensions of the reflector in Table 6.4.1.3.2.

Table 6.4.1.3.1: 44mm LED Lens Size Dimensions

| Height | Lens Diameter | Inner Diameter | Edge Thickness | Edge Width |
|---|---|---|---|---|
| 19mm | 44mm | 40mm | 3mm | 2mm |

Table 6.4.1.3.2: 44mm LED Reflector Size Dimensions

| Height | Diameter | Bottom Size |
|---|---|---|
| 11mm | 50mm | 26 x 26 mm square |

The 44mm LED lens and reflector collimator comes with a fixed bracket shown in Figure 6.4.1.3.1. This will hold the COB LED chip in place to prevent the LED moving which is especially crucial during bumpy terrain or fast movement.

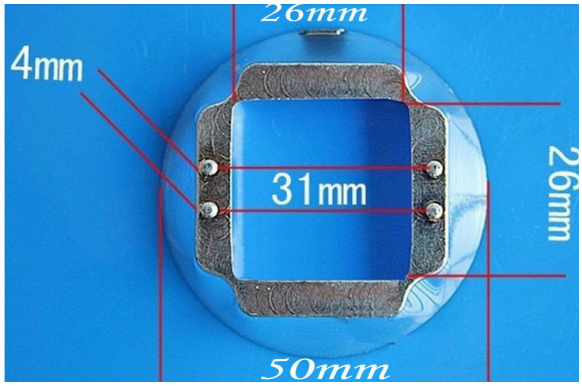



Figure 6.4.1.3.1: Integrated LED Fixed Bracket Size Dimensions

The overall product, which is available on Ebay, costs $2.07 and comes with three pieces that will be suitable for the 20W COB LED that was chosen. The product comes with three pieces as discussed previously which include the 44mm lens, reflector collimator, and a fixed bracket for the LED chip. All of which are shown below in Figure 6.4.1.3.2.

Figure 6.4.1.3.2: 44 mm Lens (bottom-left), Reflector Collimator (top-left), and Fixed Bracket (right)

## 6.4.2 Tail light and Signal Lights

The lighting system will include the tail light and signal lights in which the design will be similar to that of the headlights but rather than having very high lumen output from the LEDs it will be relatively smaller to at least 100 lumen. As discussed in section 3.4.3, the tail light will output a red LED color while the signal lights will be indicated through yellow blinking LEDs. The taillight and brake light are combined in such that the taillight will constantly be blinking red light while the brake light will turn on only during braking and be brighter than that of the taillight. To design this the parts needed will be discussed.

#### 6.4.2.1 Tail Light COB LEDs Specifications

To start off with the design of the tail light an LED is required to emit the lighting. The COB LED chosen will be capable of emitting red light as well as be dimmable. It's also important to note that the lumen output needs to at least be 50 lumen. The product found that matches all of these qualities was found on aliexpress and the product has a variety of options to choose from.

The COB LED model is called the LC-3R-C42, shown in Figure 6.4.2.1.1, and emits a red color. It costs $1.00 and the necessary voltage for this model is 2.0 - 2.6V while the current required is 600 - 700mA. It is a 3 Watt model that produces 80 - 120 lumen. The plan for this chip model to be utilized as a tail light and brake light is to vary the voltage such that as a tail light it will emit the minimum lumen, which is 80 lumen. However, the brake light will output 120 lumen once activated. The blinking for the tail light will be controlled using the microcontroller in such that the microcontroller will vary the frequency quick enough that it will quickly turn off and on the LED to create the blinking effect.


Figure 6.4.2.1.1: LC-1R-C30 Model

Compared to other counterparts, the manufacturer provides a table to easily distinguish the difference between each other [[60]]. This table is shown below, Table 6.4.2.1.1.

Table 6.4.2.1.1: Difference Between Each Red LED Model

| Model | Power | Voltage | Current | Lumen | LED Chip |
|---|---|---|---|---|---|
| LC-1R-C30 | 1 Watt | 2.0-2.4 V | 300-350 mA | 40-60 lm | 30x30 mil |
| LC-3R-C42 | 3 Watt | 2.0-2.6 V | 600-700 mA | 80-120 lm | 42x42 mil |
| LC-5R-C42 | 5 Watt | 2.0-2.4 V | 600-700 mA | 180-220 lm | 42x42 mil |

The table also points out that between the LC-3R-C42 and LC-5R-C42 the 5R is more efficient in that the required voltage is lower than the 3R model. The current remains the same between the two meaning the 5R is more efficient since it consumes less voltage and produces almost twice as much lumen output. However, the 3R model was chosen simply because the 5R model is overkill for what the team needs. The 5R outputs 180-220 lumen which is far too bright for a tail light and brake light to the point that it would be dangerous to use.

#### 6.4.2.2 Signal Lights COB LEDs Specifications

Yellow COB LEDs will be utilized for the signal lights. The bicycle rider will select the direction they wish to turn and select the option on the controller which will then turn the designated signal lights on, which will flash yellow until the turn is complete meaning until the rider turns off the signal lights. The lumen output will at minimum be 50 and of course be yellow.

The perfect choice that fits all the conditions was found at aliexpress just like the previous product however it's a different model that outputs yellow rather than red. The specific model the team is going with is the LC-3Y-C42. The LC-3Y-C42 costs $1.00 and is a 3 Watt yellow COB LED. The specific model requires 2.0 - 2.6V and 600 - 700mA. The amount of lumen that is outputted from this model ranged from 80 - 100LM, which is more than enough for the application. The other choices are shown in Table 6.4.2.2.1, provided in the manufacturers website [61].

Table 6.4.2.2.1: Difference Between Each Yellow LED Model

| Model | Power | Voltage | Current | Lumen | LED Chip |
|---|---|---|---|---|---|
| LC-1Y-C42 | 1 Watt | 2.0-2.4 V | 300-350 mA | 40-60 lm | 42x42 mil |
| LC-3Y-C42 | 3 Watt | 2.0-2.6 V | 600-700 mA | 80-100 lm | 42x42 mil |
| LC-5Y-C42 | 5 Watt | 2.0-2.4 V | 1200 mA | 200-250 lm | 42x42 mil |

The LC-3Y-C42 was selected because it's the sweet spot between the LC-1Y-C42 and LC-5Y-C42 models. The amount of lumen specifically is what brings the model the attention from the team members. The 1Y model is sufficient however we wish to provide more lumen to have a higher chance for indication of turning, which is crucial. The 5Y model provides too much lumen and would be a bad selection because of it, hence why the 3Y model was selected.

Two LC-3Y-C42 LED models will be used for the signal lights. One left of the tail light and one to the right of the tail light.

#### 6.4.2.3 Aluminum Cooling Heatsink Specifications for Tail light and Signal Lights

A heatsink is required for both tail lights and signal lights. Rather than buying 1 aluminum piece for each type of lights the team will proceed with buying a large aluminum heatsink with dimensions of 200x100x18mm, shown in Figure 6.4.2.3.1 supplied by the manufacturer website [59] By buying a large heatsink with those dimensions the tail light and signal lights will all be able to fit under one heatsink.

Just like for the headlights, an aluminum heatsink is crucial to preserve the LEDs health in regards to its brightness and life. Based on the dimension sizing, it will be placed behind and underneath the seat of the bicycle. The LEDs will then be placed on the heatsink since the size of the heatsink is enough to hold the amount of LEDS as well as the fact it can handle up to 24W max [59]. The total amount of wattage the LEDs will be producing is 3W from the headlights and two 3W from the signal lights which totals up to 8W. So, the 24W is more than enough for all LEDs to be placed on. One note about this product is that it's from the same manufacturer that is selling the 100x100x18mm version discussed for the headlight heatsink.

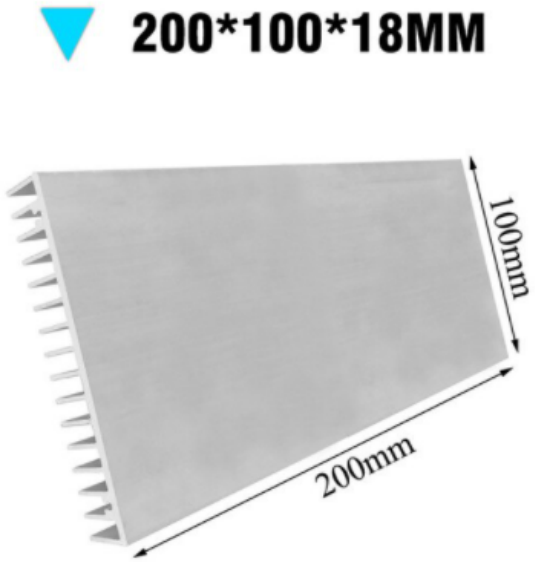



Figure 6.4.2.3.1: Dimensions of the LED Aluminum Heatsink for the Tail light and Signal Lights

#### 6.4.2.4 Lens Specifications for Tail Light and Signal Lights

The lens that will be used for the tail lights and signals lights will support 120 degrees of the light beam angle. Ideally the angle should be as high as possible to indicate braking and view of the rider better from the perspective of people behind the rider. A 10 piece set was found on Aliexpress that supports 120 degrees view angle as well as lenses suited for 3W LEDs. The dimensions of the lens is shown in Figure 6.4.2.4.1 provided by the manufacturers website [62]

## 6.5 Switches Design

The switches, which are discussed in section 8, are used to control the left and right turn signals in addition to the app. These two buttons are located on the top of the left and right handlebars, and correspond to their respective turn signals located on the back (it should be noted that the buttons connect to the microcontroller and not the lights themselves; this is of course the case because the lights would act as microcontrollers if this were possible). The purpose of having these buttons in addition to the app is for ease of use by the user. It may be considered by some

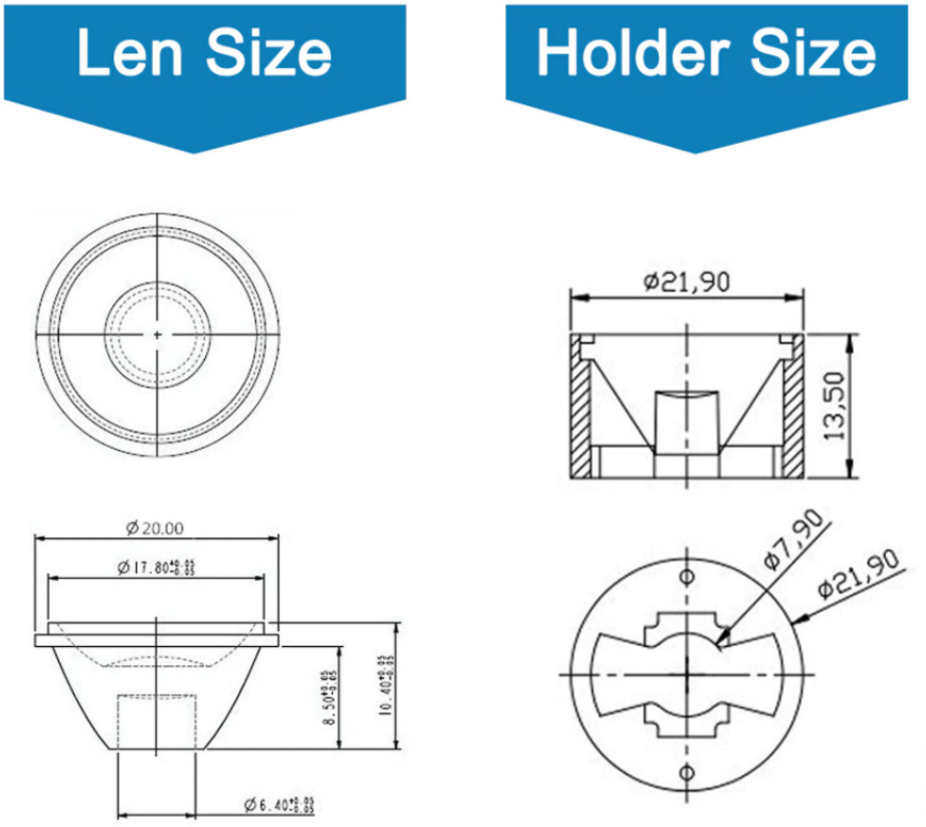



Figure 6.4.2.4.1: Dimensions of the Lens Size and Lens Holder Size in mm

users that removing your hand from the handlebars to press a button may be distracting, which is addressed in the health and safety constraints (section 4); providing buttons to users offers three benefits: ease of use, simplicity, and similarity in design to comparative products. Ease of use means that pushing the button can be done while the user's eyes are focused on their surroundings. Simplicity meaning that, in connection to ease of use, the design is similar to turn signals on a car, which can be done without noticeable need to focus on it. Finally, it can be seen on similar products (both currently for sale and in other student-design projects) that handlebar turn signals are common to the overall design scope; the argument could be made that placing turn signals on the handlebars of a bicycle is an 'obvious' first design choice.

The switches are made from common electronic buttons, which contain unpowered passive elements that, when pushed by the user, create (or complete) a circuit, allowing for a signal to be sent.


Figure 6.5.1: TWIDEC electronic button [63]

The button is attached by two pins (seen in the figure above) that connect to two wires that feed into the microcontroller. The ports of the microcontroller are selected, and the programming software is used to control the input signal that pushing the button sends. For microcontroller programming, the use of push buttons drives a high or low signal that can be inverted at the designers whim to suit the need. Then, a separate variable is created that drives the output (that is, the input button controls a variable within the microcontroller that can be related to any connected output). The outputs are the rear signals, with each controlled by a separate button (and variable, etc). There are multiple variations that can occur with a microcontroller button: one variation is that the button becomes pressed and is released, which controls the variable until the button is pushed again. Another may be a timed variable where the button pushing activates the signal for a certain duration and then deactivates, with the button in its original placement. Yet another is where the button is pushed and stays depressed until the user releases it by pressing again ('unpressing'). The TWIDEC button selected is specifically a "momentary push button", meaning that without user interference it will always set itself back to the original 'up' position. This can be considered a design constraint, and one that limits the options of how the switches can be activated; however, this also allows the designers to better choose the response of the microcontroller. It should be mentioned that in vehicles, the turn signal is released after the car has performed a wide-enough turn; this occurs when the steering wheel has been returned to center after being moved off-center. In the case of the bicycle, there is no such method that is able to know that a turn has been made in such a simplified manner. Anyone who has operated a bicycle knows that it is possible to turn both by turning the handlebars and by leaning in the desired direction. Without a more complex device such as an accelerometer this cannot be determined. Therefore, the best method would be a timed-delay signal in which pushing the button activates the respective turn signal, which then turns off by either repressing the button or by waiting some time (e.g. 10 seconds).

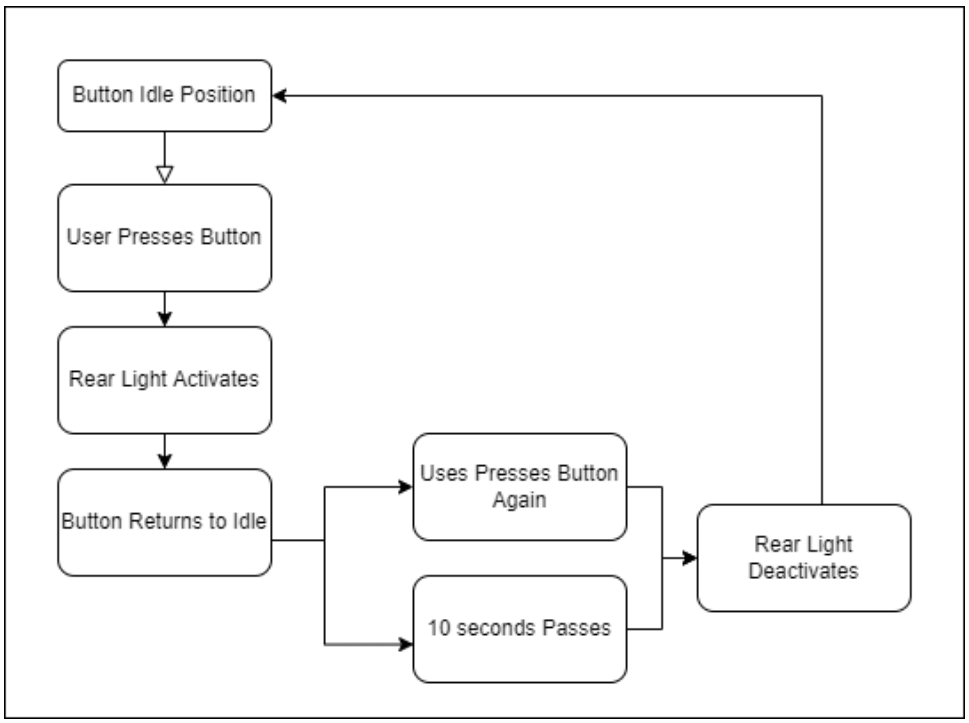



Figure 6.5.2: Flowchart for Button activation clarifying button idle position

## 6.6 Sensor Design

As discussed in the research section of this report, specifically in section 3.7, the sensors utilized for the anti-theft detection particularly for the alarm to set off. The LDTM-028K mentioned in the design section can function as a vibration sensor or an accelerometer depending on how it is set up.

### 6.6.1 Accelerometer

An ideal location for placing an accelerometer is to place it where the center of mass is for an object. Compared to the mass of the bicycle, the mass of the will be negligible, meaning we just have to find the center of mass for the bike. Bikes come in various designs so an estimation must be made about the center of mass. However, if we decide to only use the accelerometer for detecting theft, then the placement of it should not matter, as we can assume the bicycle is in an equilibrium-like state. But, if we decide to use it for other safety features like some of the mentioned stretch goals: having an SOS system where the module detects traffic collisions or other incidents, then the system cannot be assumed to be in a quasi-equilibrium state for the accelerometer [64]. Hence, it is crucial that we place the component where the angular momentum is ideally zero, or in a realistic sense, close to zero. To do so however, various calculations are required.

One of the major problems in doing so is having to use an average value for the mass of the rider since that should be accounted for by calculating the center of mass. After all, the bicycle is not

going to drive itself. It is not feasible to have the mass of the person riding be measured every time, various levels of tolerances for the activation of triggers and estimations of expected body masses (plus any additional mass carried by the body like in backpacks etc.) corresponding to these tolerance levels must be made. Fortunately, bicycle designers do a lot of research on this, which requires more detail and more resources, making this part somewhat easier to accomplish. Another strategy that can be used alongside with the former is having a recommended placement chart that would minimize the angular momentum for various body types.

## 6.7 Software Design

This product's software is composed of the MCU operating system along with the mobile application. The software functionality per part includes sensor input handling, data evaluation, and light system functionality for the MCU and bluetooth-interfaced alarm and light control for the mobile application. Figure 6.7.1 depicts all of the necessary inputs and outputs for each part.

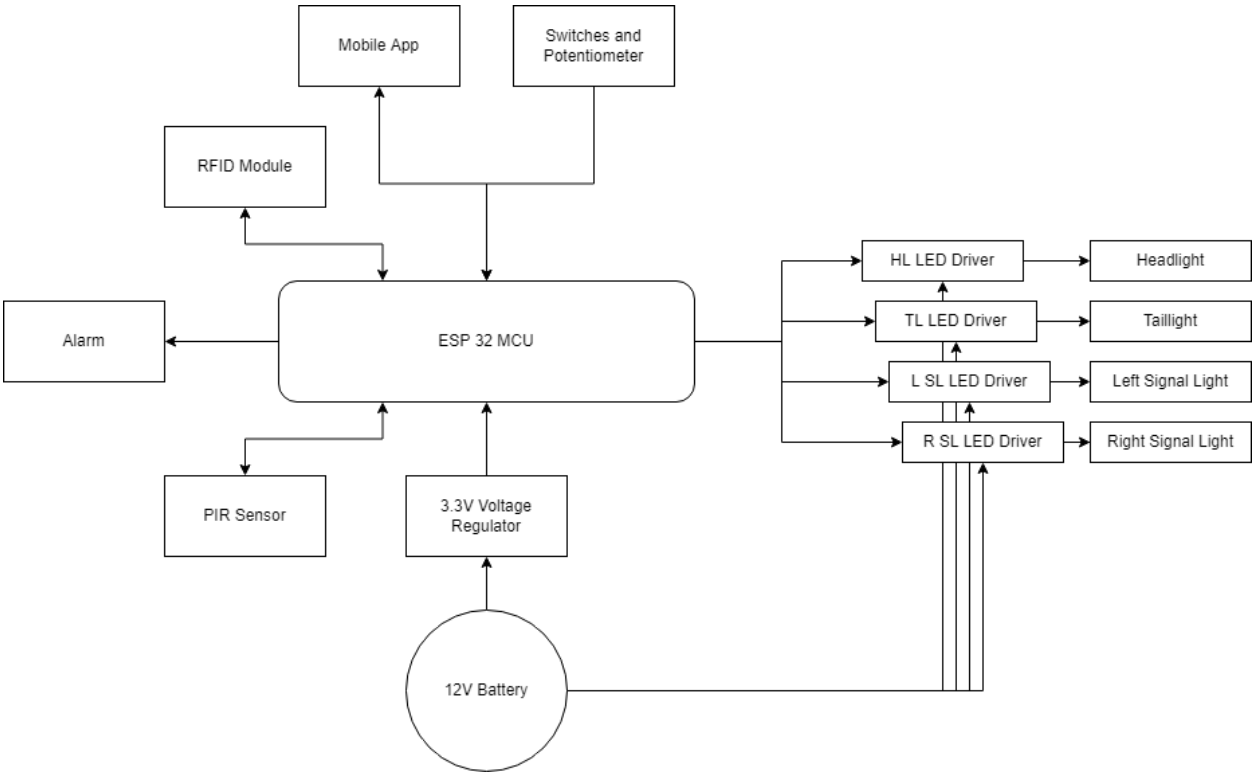



Figure 6.7.1: Software Design Block Diagram

This product does not require a lot of code and it should thus be possible to condense all necessary functions into two respective classes for our main product components. The largest class will likely be Application, shown below in Figure 6.7.2. The Application class will consist of quite a bit of UI along with bluetooth functionality—detailed in Section 6.7.2—while the Controller class will handle direct physical inputs/outputs along with bluetooth functionality.

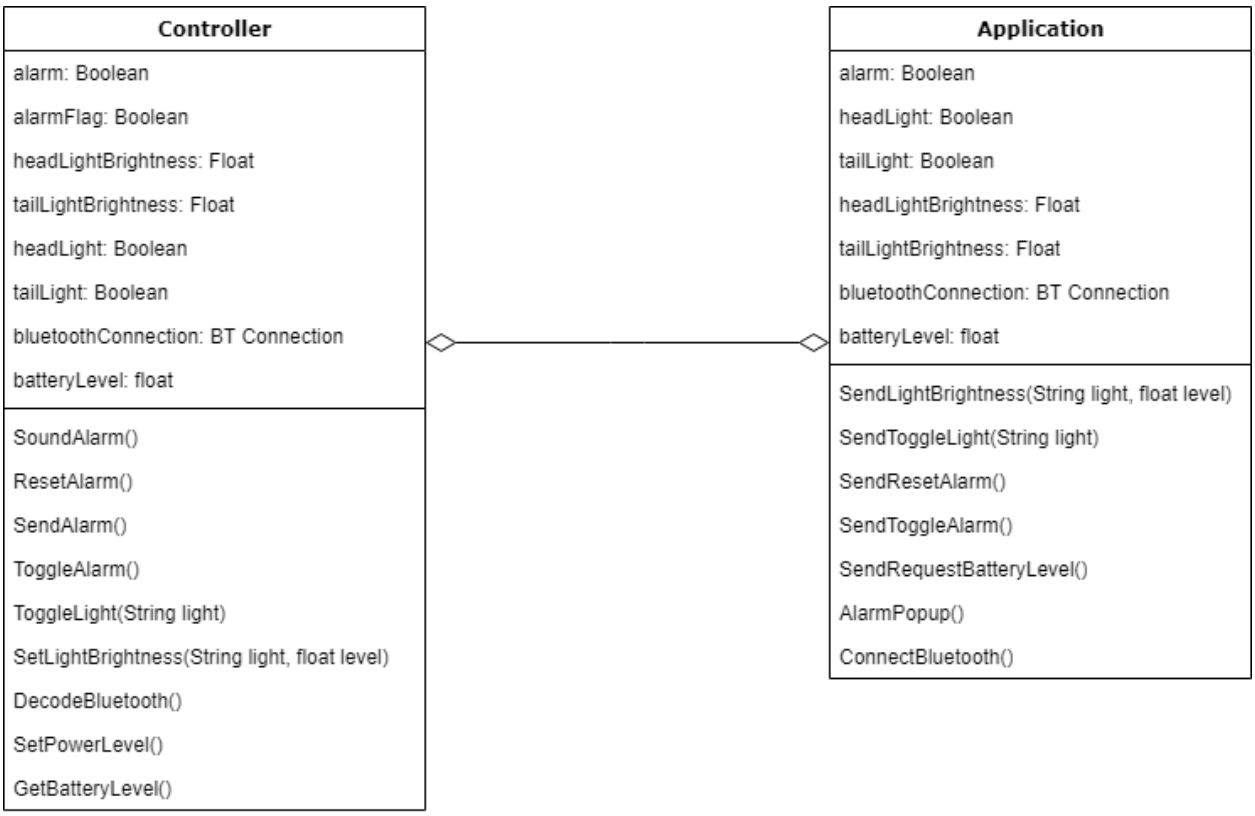



Figure 6.7.2: Software Class Diagram

### 6.7.1 MCU Software

The MCU OS must be able to support control via physical input and the bluetooth-connected app. The MCU will have inputs including controls, sensors, and bluetooth; its outputs will be bluetooth, alarm triggering, turn signal activation, and head/tail light luminosity plus activation. These functionalities should be easily controllable within a method.

Alarm functionality will consist of four methods: SoundAlarm which will be executed if the alarm is activated and one of the sensors is tripped; Reset Alarm which will be executed once the user has pressed the Reset button detailed in Section 6.7.2; SendAlarm which will be executed if SoundAlarm is executed and the user is still connected, it will notify the user that the Alarm has been tripped; and ToggleAlarm which will turn the alarm on and off.

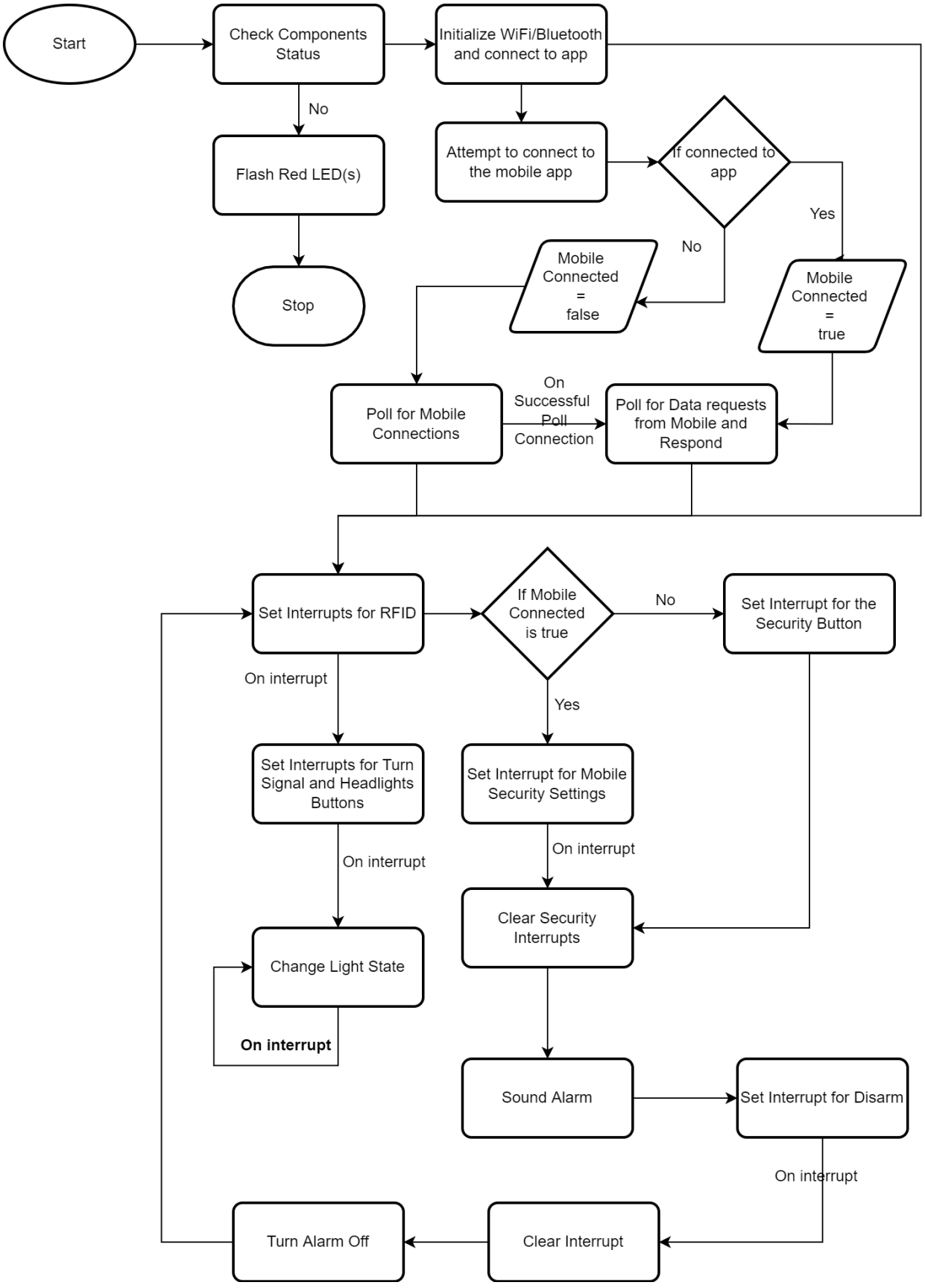
Lighting functionality will consist of two methods: ToggleLight which will take a string Light as input, this input will then be decoded into Left or Right Turn Signal or Head or Tail Light. This method's intent is to turn the lights on and off. In the case of Head or Tail lights this function will act as a simple switch. In the case of the turn signals this function will still act as a switch but will make the turn signals flash. SetLightBrightness will take a string Light and float Level, this function will only work for the Head and Tail Light and is the functionality for the UI's sliders.

The Bluetooth functionality will consist of two methods: SendAlarm which was already mentioned and DecodeBluetooth which will run necessary functionality after receiving and decoding com-

mands from the mobile application. Bluetooth-received commands include SetLightBrightness, ToggleLight, ToggleAlarm, and ResetAlarm.

Lastly we have Power functionality which will also consist of two methods: SetPowerLevel which will deal with Low Power Modes for energy efficiency; and GetBatteryLevel which will be called on by DecodeBluetooth to send the mobile app the current battery level.

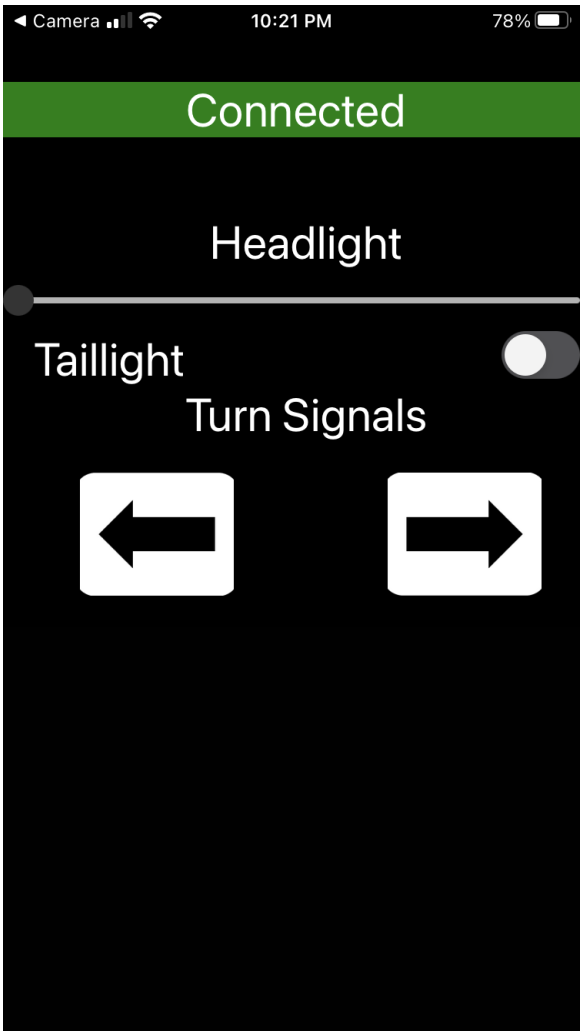
Figure 6.7.1.1: Microcontroller Software Control Flow Design



### 6.7.2 Mobile Application

The mobile application, herein "app", must be able to support touch-based controls and communication over WiFi. Inputs include touch, outputs include WiFi. This application must support iOS and Android while providing a unified appearance.

Figure 6.7.2.1: Mobile App UI Design




Control functionality will consist of one screen, four controls, and various methods. Figure 6.7.2.1 (a) is the screen that will greet the user after they have connected to their device. This screen will also become the startup screen after the user initially connects to their device. The circle beside each of the Taillight button lets the user know if it is enabled with the darker circle representing Enabled and White circle representing Disabled. The four controls include Headlight, Taillight, and Left  Right turn signals, each able to toggle their respective part or adjust light level. The four methods are SendLightBrightness which will be activated by the light sliders and will send the respective light level to the MCU for adjustment and SendToggleLight which will take a string Light and will send the respective activation signal.

Turn signal activation will be represented on the app by a flashing button. As shown in Figure 6.7.2.2, the respective button will change from a gray background and black arrow to an orange background and yellow arrow. This will flash at a specified interval to best emulate the actual signal flashing. This design is deprecated by Figure 6.7.2.1.

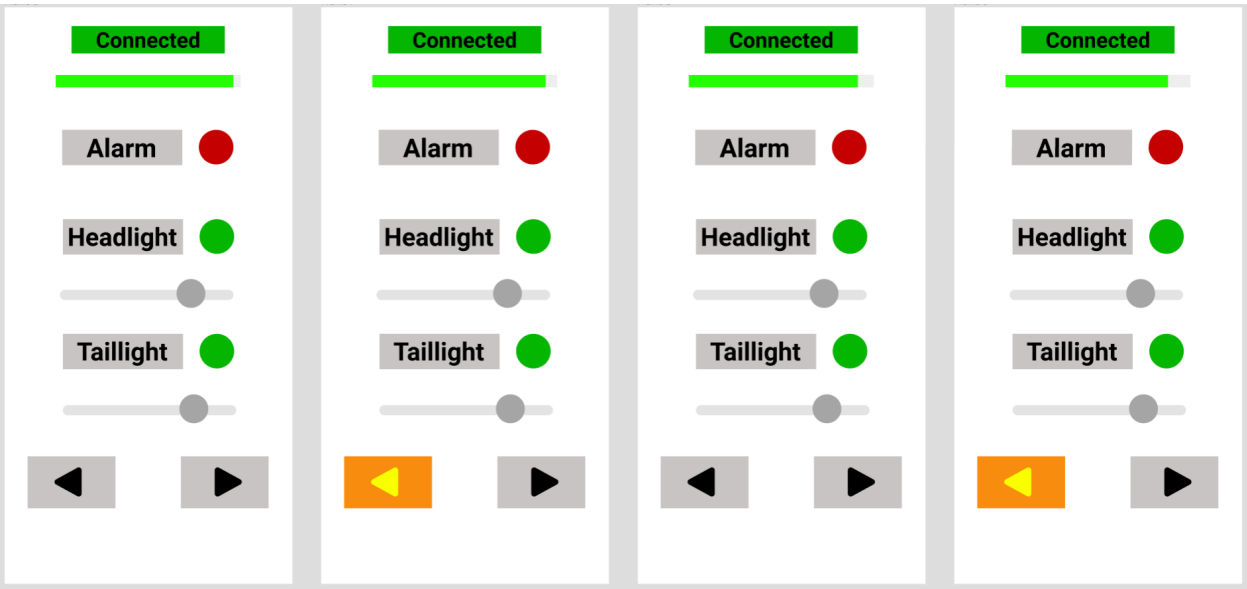



Figure 6.7.2.2: Control Screen - Turn Signal

### 6.7.3 Events

Both the mobile application and the MCU software will use various techniques to collect and request data, and respond to requests and take actions. RTOSes typically have out-of-the-box support for working with and handling events. In basic microcontrollers and simple programming environments that do not use specialized hardware, events can surely be handled. They are however, a lot more restrictive. Therefore, the microcontroller we leaned toward had many options that would let us handle events. Mobile devices which run full-blown operating systems with support for event handling hardware.

In the microcontroller flowchart, it can be seen that there are many event handling being done, mostly through interrupts. The ESP32 support a large number of interrupts which allows us to overly rely on them. We will be using polls for creating connections to the mobile, and interrupts for RFID state changes, security button pushes, mobile interactions, headlight buttons, turn signal buttons and the brake pushes.

#### 6.7.3.1 Polling

Polling is one of the easiest ways to listen for and fetch events. However, it is not very efficient if the hardware does not support it. Moreover, if the processor is single threaded, then the polling events are going to be blocking, meaning that either other tasks cannot be done when the polling occurs or there has to be some sort off concurrency system to allow multiple tasks to run at the same time. Considering how most systems take and do more than one task, even if they are low

powered singly threaded systems, they support running multiple task at the same time. While polling may seem as a bad idea, it has its use cases and is definitely not inferior to the alternatives. In fact, USB keyboards, which most people have and use wih desktop computers right now, use the polling method to get the keyboard input events. This is in stark contrast from the interrupt based system PS/2 keyboards, introduced by IBM, that were once popular had. Polling will be used to fetch some of the data from/to the microcontroller and the mobile application although not to the extent of interrupts in the microcontroller.

#### 6.7.3.2 Interrupts

Interrupts are very useful to handle events and just working with them. Interrupts are relatively more difficult to work and complex systems like operating systems discourage the use of interrupts. However, we will not be using an operating system for our microcontroller software. Interrupts offer a lot of flexibility when designing software, especially they can be prioritized (somewhat) and can be entirely disabled with hardware flags. Depending on the kind of environment, events can either preemptively be scheduled or non-preemptively be scheduled, similar to the multitasking strategies used in operating systems. Interrupts have low overhead compared to alternatives however this can be abused by malicious actors to ruin the system or even uninformed actors who do not act out of bad faith. So proper care must be taken to ensure that interrupts work as expected.

# 7 Prototype Testing

## 7.1 Hardware Testing Environment

The testing will take place in multiple: one being the Senior Design Lab located in the Engineering building at UCF campus, and the others being various places on and near UCF campus. The Senior Design Lab has access to advanced equipment including, but not limited to, function generators and oscilloscopes that would other wise be financially unavailable to researchers. Because of the size of the project, the electronic hardware will be tested independently of the bicycle and chassis originally, using the electronic testing equipment available. The tests conducted will ensure that the design specifications given by the manufacture's are accurate to a degree that will not interfere with the project, and also to make sure that hardware connections do not create any unforeseen problems. Handling these electronic tests prior to chassis assembly and connection to the bicycle ensures that troubleshooting progresses in a way that is logical and convenient for designers; if any significant problems arise later, the entire model, bicycle included, can be brought into the lab as well, this is non-ideal but still possible during troubleshooting steps. The specific models of the electronic equipment used will be accounted for during the testing stages. Other than the Design Lab, the UCF campus offers access to large sections of pedestrian-only sidewalks as well as medium-traffic roads with both bike lanes and sidewalks. These bike lanes and sidewalks will be used during conduction of the mobile LED tests. Naturally, the safety of the researchers, both on and off the bicycle, will have to be properly accounted for during tests, so choosing an adequate road that is not over-populated with vehicles will be done. In addition to sidewalks and bike lanes, UCF campus has numerous bicycle-rack locations of various size that offer places to test the alarm system in a real scenario location.

## 7.2 Hardware Testing

As detailed in section 6, the hardware will be tested in two stages. The first stage will be a simple test done by the designers to see if the project works as intended, e.g., if the turn signals can be seen, if the alarm sounds properly and can be heard, etc. Once it is confirmed that the hardware works properly, the specification of the hardware can be tested using various means; there are a multitude of apps that are able to give lumen and decibel readings which act as free hardware testers. Naturally, if the results seem significantly off what should be expected, then more 'professional' equipment can be acquired and used; this equipment is usually confirmed to adhere to modern standards. On another level, the output rating can also be tested against the expected value based on manufacturer specifications and calculations done by designers. When discrepancies occur, there are several back-ups that can be turned to.

Another hardware test that is less in scope of the project, which is discussed in section 8, is a general test of strength and resilience to damage as well as sustainability; the chassis is meant to

mitigate (ideally remove) any jostling that can occur due to improper securing of the chassis to the bike. Another test of general resilience can be to make sure the device does not break due to minor falls; that is, small collisions that may occur to common cyclists.

### 7.2.1 Microcontroller PCB Design Testing

The testing of the Micro controller PCB Design is the test that will be performed outside of the chassis (i.e., apart from the bicycle) when possible and will take place in the Senior Design Lab, mentioned above. The micro controller will be tested when connected to all peripherals (LED, speaker) and the testing will go from a broad to a more refined scope. That is, the first test to be done will be to ensure that the connections made are all secure and are not effected by any problems that could occur with little or no human interaction (e.g., ensuring that the connections between electronics are not loose enough to become disconnected due to simple movements, and that the connections to the peripherals are constant, not fluctuating in intensity). Once this is ensured, the electronic testing equipment can be used to make sure that the specifications by both the manufacturer and the researcher agree with the expected measured results. If there is a discrepancy, then an even more refined test must take place, but if there is no such discrepancy, the software tests can be concluded (assuming no problems emit later during design).

### 7.2.2 Lighting System Testing

As mentioned previously, the lighting for the tail lights must reach a level of at least 75 Lumens during the night. The testing for this will be done at multiple points during the night and at multiple distances. The testing will also be done during the day, however, there must be a slightly different criteria, as the daylight will affect the readings. During the daytime, that is, between sunrise and sunset, the lighting will be tested when the bike is idle and moving, and the use of both turn signals will be viewed by researchers as well as filmed (for comparison and for use in the final video demonstration). If the rear turn signals can be seen, with little or no doubt of visibility, then the lights can be considered properly visible; the lumens during the daylight test will be included to ensure that there is data for this test. The nighttime test will be similar to the daytime test, and will take place between sunset and sunrise. At multiple times during the night, e.g., 1 test every hour until 5 tests have been done, the LEDs will be tested using a LUX measurement tool. Both the idle and mobile tests will be done in an area common for bicyclists to ensure data is reflexive of the scope of the project. The nighttime test will be done at multiple distances to show the 'drop-off' of visibility as the distance between source and viewer increases, which should correlate to the inverse-square law of light intensity.

### 7.2.3 Anti Theft Alarm Testing

The theft alarm testing will be done to test if the alarm sounds during the supposed theft or vandalism done to the bicycle. During the testing, the hardware of the project will be secured in a way that is reflexive of the final version (that is, also in a such a way that allows for troubleshooting, while still being secured properly). The software will be properly set so that the alarm is 'primed' and one of the researchers will attempt to interfere with the bicycle while it is idle in such a way that might mimic its theft. During this time, the event will be filmed for

researcher use, and the decibel rating will me actively measured at a distance of about 8 meters. The bike will be agitated in a way that shows that the bike is being moved with the intent to steal or relocate it, at which time the alarm sensor and sound level will be tested for success. If the alarm sounds, and the decibel level is at or above the expected/desired level, then the alarm will be deactivated properly in a way that a user of the product would. If the the alarm is properly activated and deactivated, then the tests can be considered successful. One caveat to the these alarm/theft tests is that it is beyond the researchers scope to test the effectiveness of the alarm on theft deterrence. This would require a highly involved psychological test that goes well beyond the knowledge and sanction of the researchers. In lieu of this, the effectiveness of other modern theft deterrence such as car alarms can be researched and referenced along side the physical data gathered from the alarm tests.

## 7.3 Software Testing

In order to ensure proper functioning of the software components of SAFE T, the software will also be thoroughly tested along with the hardware. There are two major software components programmed each of which will be tested using the most ideal testing techniques for their respective design models. The testing will not guarantee the detection and removal of all errors, as testing cannot evaluate every single execution path a program will take. Despite this, testing strategies are very much preferred to minimize the time involved in debugging software problems.

We will be hosting our code on a git supported platform like GitHub or GitLab, with GitLab having the option to be self-hosted. These platforms support tools that can integrate well with testing software and most importantly, makes it easier to test collaborative software, like the ones written for this project.

### 7.3.1 Module Software Testing

The MCU requires interaction with the hardware to effectively be tested. There are solutions that can emulate changes in hardware. However, this will not thoroughly detect any flaws with the hardware connections so a model that can prevent these problems will be required. Embedded Testing is a good way to ensure both the software and hardware work together. This cannot be automated software, mostly because the physical interaction from the hardware is difficult to automate. Fortunately, they do not take too long to test as the compilation times are shorter and the dependencies are extremely small.

The endpoints for the mobile app, however, can be tested using an automated system. Depending on the protocol or the API specification used, API testing tools can be easily configured to automate tests that do not rely on the hardware inputs from the module.

### 7.3.2 Mobile Application Testing

Unlike the code for the embedded environment, the application code requires a lot more time to compile due to the large number of dependencies and differences in target architecture etc., hence manual testing is not feasible. The developers of these platforms recognize this and so

mobile software development tools are typically packaged with libraries for unit testing and related testing methods. No matter what testing method is used, the goal is to have them automated at every update for quick debugging of code.

#### 7.3.2.1 Unit Testing

Depending on the language used, unit testing libraries available differ in type and style. However, the fundamental idea remains the same. Unit testing is essentially testing every unit of code that can be tested to see if the function works as expected and therefore is good to be used. A major benefit unit tests add is to prevent deployment of breaking code. But unit testing acts more like a sieve than a barrier for errors, meaning not every error will be caught, as mentioned before.

Unit testing helps prevent code while the application is being developed as long as the tests are written for the application along with the application itself.

#### 7.3.2.2 Integration Testing

Integration testing is the step after unit testing. While unit testing involves testing every unit, integration testing involves the integration of the units to their modules and tests the various modules on their own. This cannot always be done as sometimes units may exist on their own, independent of modules.

#### 7.3.2.3 Code Analyzers

The mobile software development tools also come with other tools that may help avoid bugs by catching errors. One of the tools that come for testing code are called linters. Linters are static code analysis tools that catch errors even before the program is compiled or executed. Although these tools are run in different ways and sometimes may even be integrated with the development environment or sometimes provided as an optional command within the environment. There are dynamic options as well for code analysis but their benefits are highly dependent on the language used.

#### 7.3.2.4 UI Testing

Like other types of testing, UI testing can also be done either manually or automatically. The manual mode of testing as usual is not reliable and tedious to do. The automated testing is more consistent and does not require programmers spending more time testing. Like the other tools, UI testing libraries are also usually shipped with the frameworks, making it easier for developers to test their application's user interface.

#### 7.3.2.5 Continuous Integration

While our team is not big to see the greater benefits of CI, the practices used can be used to automate many things and do many things as soon as the code is written to the main "copy" of the code. We will be using a version control system (most likely git) to keep track of all the changes we make. CI also involves using build automation tools. Additionally, the code will be

tested on deployment, the tests are going to be the same as the ones mentioned before: for unit and integration testing, except with code to automate them.

We do not intend to make this fully fledged as implementing automated tests and works can be very time consuming and the benefits are not as great for a small team. Considering how the development cycle of engineering products is limited, it is not feasible to implement a full scale system. However, we will be using various tools freely available from vendors that make it easier to automate most things.

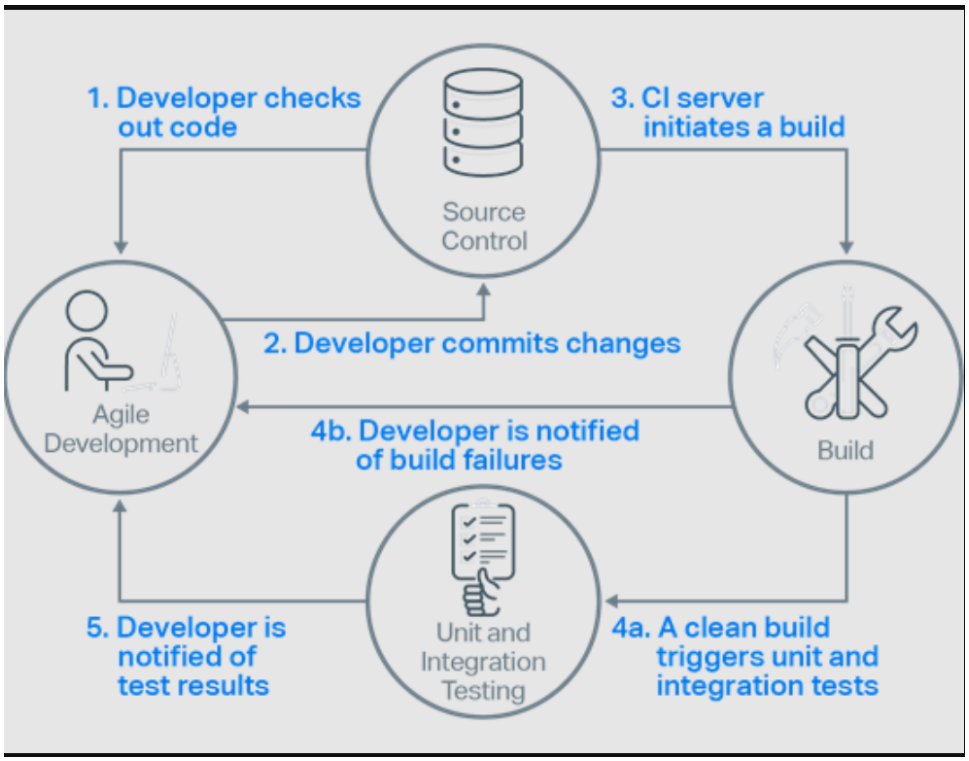



Figure 7.3.2.1: Continuous Integration

# 8 Design and Engineering Specifications

In this section, the physical design constraints will be detailed. The design draft will be given, both in description and preliminary model, and the parameters that govern the design will be detailed. The use of preliminary models are used as placeholders for the final design; the final design will include CAD renderings that are used as an additional reference during the building stages.

## 8.1 Handlebar Turn Signal Connection

The turn signals, which are one of the focal points of the design, are connected to the ends of each handlebar. The left- and right-side handlebars each hold an electronic button that is connected to the microcontroller by wires, which correspond to the left and right turn signal located at the rear of the bicycle.

The buttons themselves are small, with the weight of each being less than 2 ounces. Combined with the wires the weight of this portion can be considered negligible.

The buttons will be attached to the ends of each handlebar with a small plastic clip that will be designed to be removed and reattached at the user's desire. However, the prototype will be held by cruder methods such as electrical tape or zip-ties. The wires will be similarly secured running the length of the handlebars into the microcontroller.

One design flaw in this is that the wires or buttons may be cut by a potential threat. While this would render the buttons useless, the alarm system would remain intact; meaning that the threat would only come from someone looking to cause damage rather than commit theft. This problem is generally unavoidable (as any exposed wire would be), however, the risk is considered low.

Figure 8.1.1: Handlebar Preliminary model

## 8.2 Rear Turn Signal Screen

The turn signals are meant to be controlled by the use of the buttons on the handlebars with a backup method located on the user's smartphone. Both of these are linked (directly and indirectly, respectively) to the motherboard (microcontroller) located at the center of the frame. The microcontroller then sends signals directly to the LED screens located at the rear of the bike by wires attached to the frame. These are two small LED screens located underneath and behind the seat of the bicycle that correspond to the left and right turn signals. Each of these LED screens meets the design lumen specifications for use in daylight and during the night.

The screens measure 2 inches by 2 inches by 0.5 inches. The screens are durable enough to survive minor low-speed falls that can occur with normal bicycle operation, but, like nearly every bicycle accessory, are not designed to withstand the impact of a collision with an automobile or a high-speed crash. These two screens will be mounted by a plastic attachment that will hold both screens on either side and attach via screws to the vertical frame bar located underneath the seat.

The screens will only display a rudimentary ON/OFF signal, either a blinking light or, if design permits, an arrow to indicate direction and better alert people and cars around the user. As with the buttons, general vandalism is a design flaw that is considered unavoidable.

Figure 8.2.1: Module Preliminary model

## 8.3 Microcontroller Chassis

The centerpiece of the design, which acts as the heart of the equipment, is the microcontroller. The microcontroller will be placed in a plastic container that will attach to the center bar of the frame; common road bikes feature two center frames: one parallel with the ground and one at an angle reaching down from the handlebars to the gear wheel. The horizontal bar is ideal of the two bars to hold the microcontroller, as both effects of gravity and jostling are less on the horizontal bar than the latter. By the diagram below, the horizontal tube is the "Top Tube" and the angled tube is the "Down Tube". The chassis will be held in place to the frame by these two main tubes.. By having this connection, the degrees of freedom of movement is reduced to near zero. The two tubes that connect to the bicycle frame will be attached with metal screws, or with a strong epoxy glue. Ideally, the chassis will not be moved or taken off during design; as mentioned in the constraints section, the chassis is a portion of the design that can be adjusted at any time and is not bound to any design constraints of the software. The entire chassis will consist of four pieces: one top shell that acts as the cover for the microcontroller, one lower shell that connects to the top shell to hold the microcontroller, and two pieces that aid the lower shell in attaching itself to the two bars (two bars on the bike make for the need of three pieces to properly attach). The rod piece that connects the two pieces that wrap around the tubes may be attached after-printing by means of strong epoxy glue; this is done due to design constraints with the CAD software.

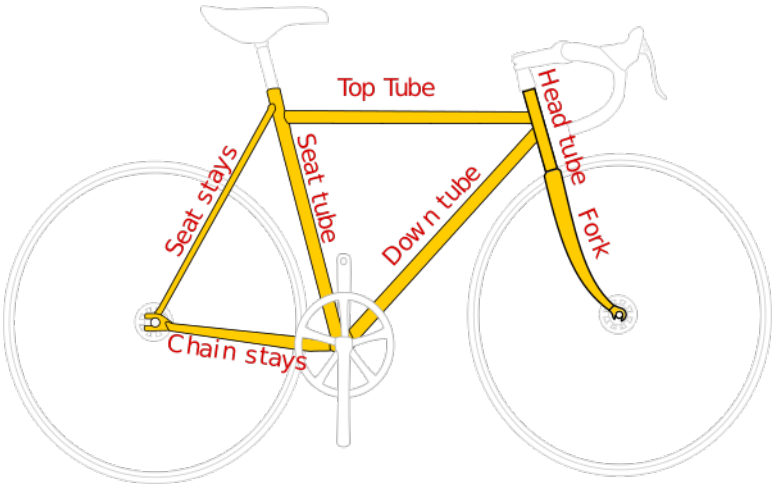



Figure 8.3.1: Road bike with common frame design

The center container can be considered the main component of the chassis, and the bar underneath acts as a physical stabilizer. The center container will sit above the top tube, and contain two pieces: the first piece will be a flat surface that sits above the bar. With a lower portion (attached to the model) that wraps partially around the top tube.

The chassis will be just wide enough to hold the microcontroller and additional electronic hardware, with small holes placed underneath in the rear and front to allow access to the wires to the buttons and LED screens. These holes will be the size of, or just above the diameter of the wires (approximately 10 AWG or 2.588mm) and on the bottom to reduce tampering and water damage.

The top two pieces of the chassis are secured surrounding the microcontroller (similar to a clamshell container). This design is implemented so that the microcontroller can be added and removed as the design process continues; at the final stage of design, the chassis may be connected using tamper-proof screws to mimic a more 'ready-to-sell' product.

Figure 8.3.2: Chassis preliminary model

The dimensions of the chassis are relevant to the dimensions of the microcontroller and relevant to the dimensions of the bicycle being used. The bicycle being used is a TREK road bike, which can be considered having a common road bike frame. The only relevant dimension of the bike is in the diameters of the two bars that the chassis will attach to. The frame-bars for common road bikes do not taper and can be considered to be generally uniform in dimensions. The top tube is commonly 1.25 inches (31.75 mm) and is accurate for the bicycle being used in design. The down tube is roughly the same size as the top tube, at 1.25 inches. It is found in some models that the down tube is larger than the top tube by about 0.25 inches, which may be accounted for in later chassis design.

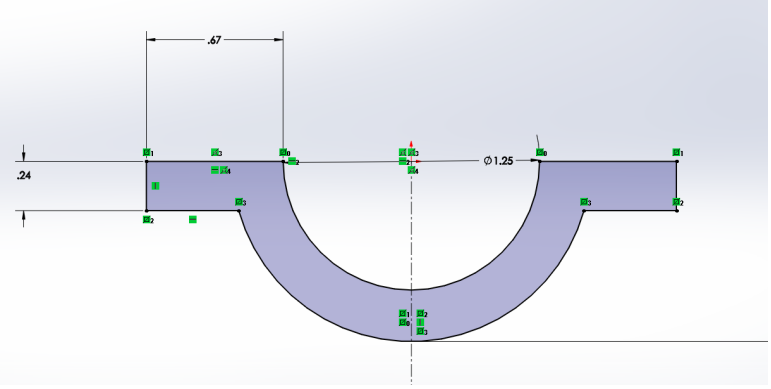



Figure 8.3.3: Side-view of the lowest chassis piece

It should be noted that in previous designs, this chassis lies at the rear of the bicycle, supported by a rear bicycle-rack. In this rear-rack design, all electronic parts sit within a box that rests on a bicycle's rear-rack (which attaches to the bolts of the rear wheel). This model design, relative to the current design, comes with both strengths and flaws. The strength in the 'rear-rack' design is in that it is out of the way of the user. The center chassis design will slightly infringe on the users range of motion in their legs, as moving toward the chassis will be restricted; this design flaw was considered, however, the common range of motion for bicyclist does not include this 'thighs together' motion, so the chassis benign significantly in the way is not considered a restrictive flaw. One flaw in the rear-rack design is that it is significantly bulkier, and most likely heavier than the plastic chassis model.

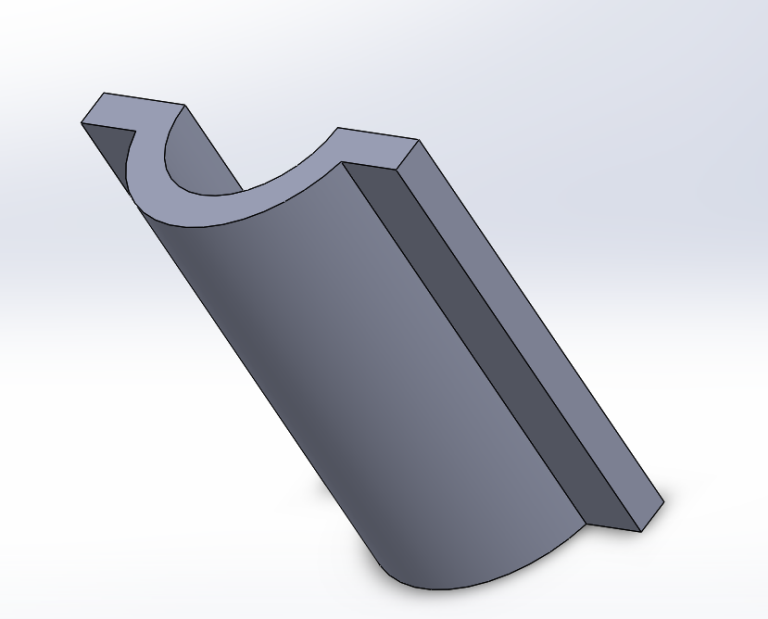


Figure 8.3.4: Extruded view of the lowest chassis piece

One thing that is a double edged sword is the use of the already-built rear rack; this is an additional piece of hardware that needs to be either owned or purchased for the electronic components to be held in place; without this rear rack the whole design cannot be mounted. In the plastic-chassis model, the rear-rack is eliminated and the mounting hardware is included in the design. However, this adds significant time and effort for the designers, which includes the need to print the pieces rather than purchase a simple bicycle accessory (for which the rear rack is). Regardless of the debatable pros and cons of the design choice, the plastic chassis model is used.

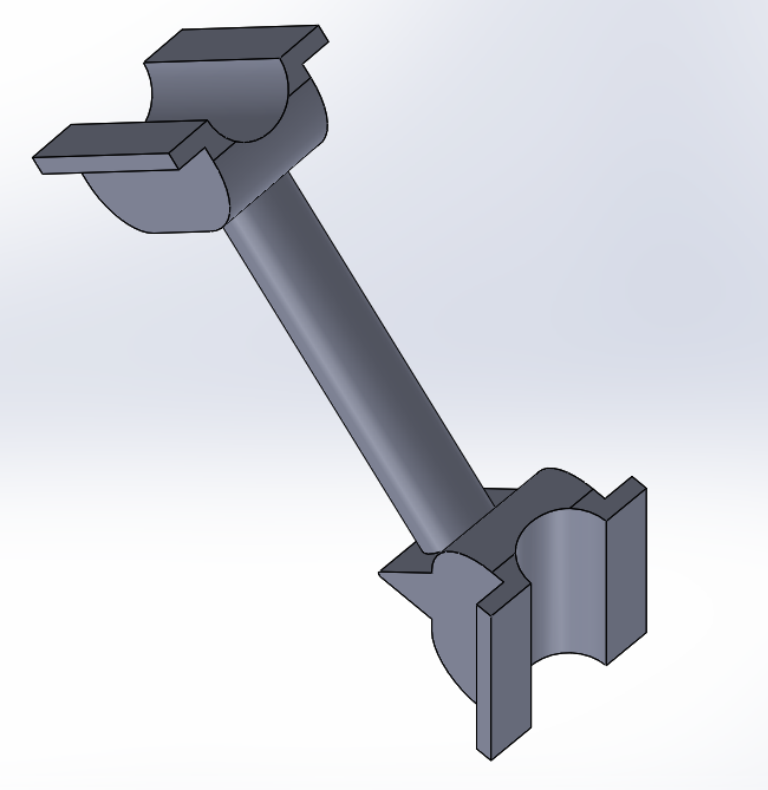


Figure 8.3.5: Connector piece shown as an assembly

The piece that connects to the very bottom of the model, which connects underneath the down tube, and is the lowest part of the model, is designed in a horseshoe-like shape. The curvature of the center is designed to wrap halfway around the down tube, with the side 'wings' used to secure the two pieces to each other. It has not been determined yet whether this connection will be held with glue or by screws; glue is the easiest solution but is naturally difficult to remove if needed. Screws can be accounted for in SOLIDWORKS and holes can be made to accommodate specific machine standards. This piece is made to have an all-around width of ¼ inches, which is found to be sufficient for stability. The piece is extruded 3 inches, which is similarly chosen to fit a sufficient part of the down tube, providing stability without being a weight or size hindrance.

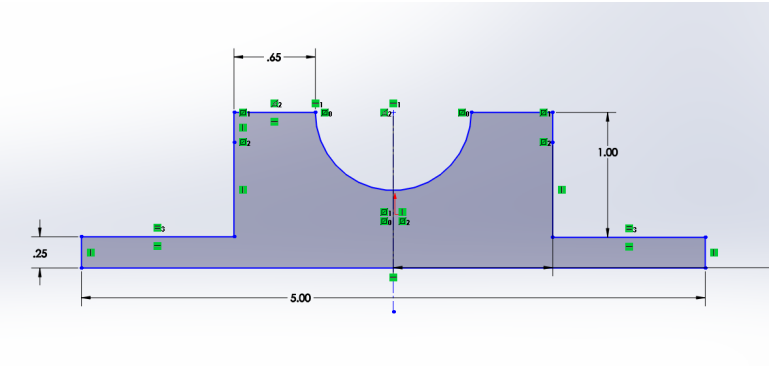



Figure 8.3.6: Bed piece side view

The piece that connects above this intended to be a single part that connects the two bike tubes with two horseshoes on either end, connected by a single cylindrical rod that spans the length of the two tubes, at an angle of about 45 degrees (this, as well as nearly all dimensions of the chassis, can be changed or edited if needed). On the following diagrams, this piece is broken into three, as the editing of dimensions is found to be easier in this form than combined. The rod that connects the pieces is generally unchanged, making it so that bringing these three into one is a simple matter of securing (gluing) the two horseshoes to each end of the rod. The rod itself is set at 5 inches, which, when factoring in the length of the two horseshoes, spans the top tube to the down tube; it should be noted that the distance from the top tube to the down tube increases due to the angle between the two, so any discrepancy of the dimension of the rod should fix itself by simply shifting the whole chassis forward or backward (i.e. toward the handlebars, or toward the gears).

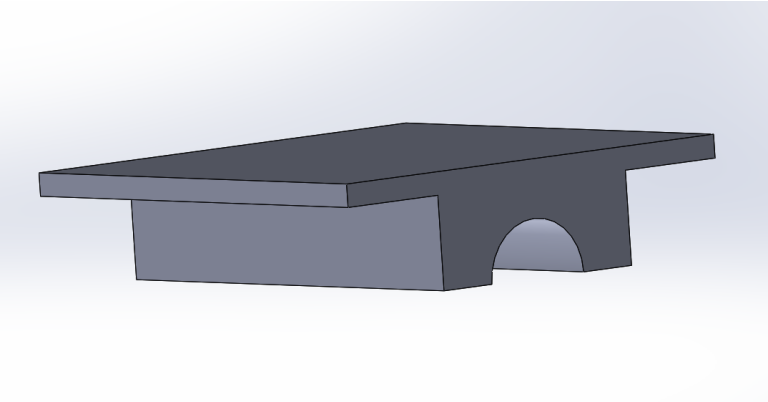


Figure 8.3.7: Bed piece extruded view

The third piece is called the "bed" and features a curve of the same diameter as the other horseshoes, so that it completes the wrap around of the top tube, and includes a flat rectangular prism in which the microcontrollers will sit. The dimensions of the bed are set to 5 inches by 5 inches, which is sized to hold the microcontroller lying face-down on the bed. As with the other pieces, there is an overall width of 0.25 inches used in the side-view design. The entire bed side-view is extruded 5 inches, which includes 2 inches of 'hang-off' material; this is included so that the bed and the shell may be shifted later in design. The bed is intended to connect to the upper portion of the connector around the top tube in the same way that the bottom piece attaches to the connector around the down tube; either by screws or epoxy glue as necessary.

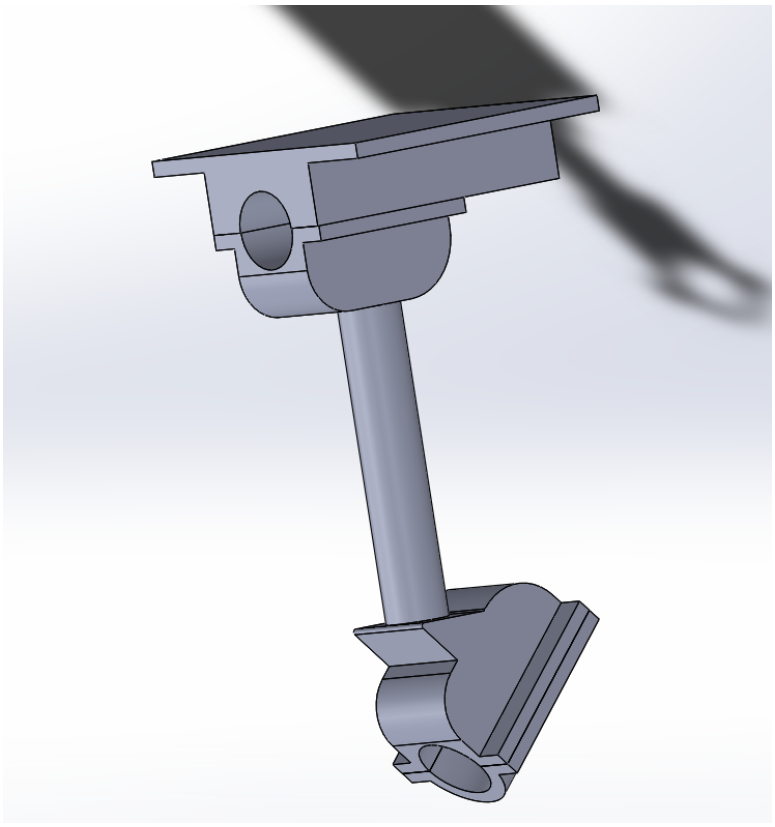


Figure 8.3.8: Preliminary version of the assembled chassis, not including the shell that attaches on top of the bed

The final part, the shell, is simply a box that sits on top of the bed and protects the microcontroller from primarily the weather and vandalism. The bed is a box of dimensions 5 inches by 5 inches by 3 inches, which is shelled (a common and simple built-in feature of SOLIDWORKS) to have walls of 0.25 inches. The shell will be attached to the bed by screws, however, attaching the shell to the bed will most likely be one of the final steps of design; as it should occur after as much troubleshooting and testing as possible, so that the shell and bed do not have to be separated more times than necessary. Because microcontrollers (and all common electronics) do not conform to 'box dimensions', the shell may be padded with non-conductive soft material such as corrugated cardboard to help avoid jousting of the microcontrollers. This design step will be considered toward the end.

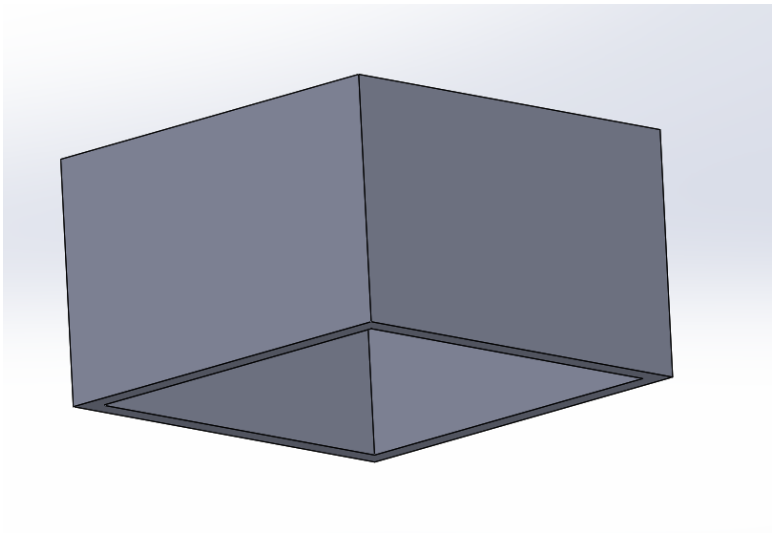


Figure 8.3.9: Extruded view of the shell piece

As mentioned under design constraints (section 4.4) the current design of the chassis does not allow for the adjustment of the plastic pieces; this is a factor that goes beyond the scope of the design and requires advanced mechanical and/or modeling techniques that the designers do not currently possess. However, the layout of the common bicycle frame allows for a 'built-in' adjustment; as the distance from the top tube to the down tube varies with distance to the handlebars. Therefore, the chassis can be adjusted until it is tightened in place.

As explained, the chassis is intended to be 3D printed, which is helpful as CAD modeling allows for precise design specification. However, it has been considered by designers that some parts may be able to be molded and /or formed using common PVC. PVC is a cheap, reasonably resilient plastic material whose shape allows for the forming over the tubes, and the connection between the tubes, to be made in the similar way as the CAD models above. In the stages of physical design, after the initial models have been printed and the chassis is being assembled, additional modifications may be made using PVC or other common materials as needed. This is only speculated in the current design stage.

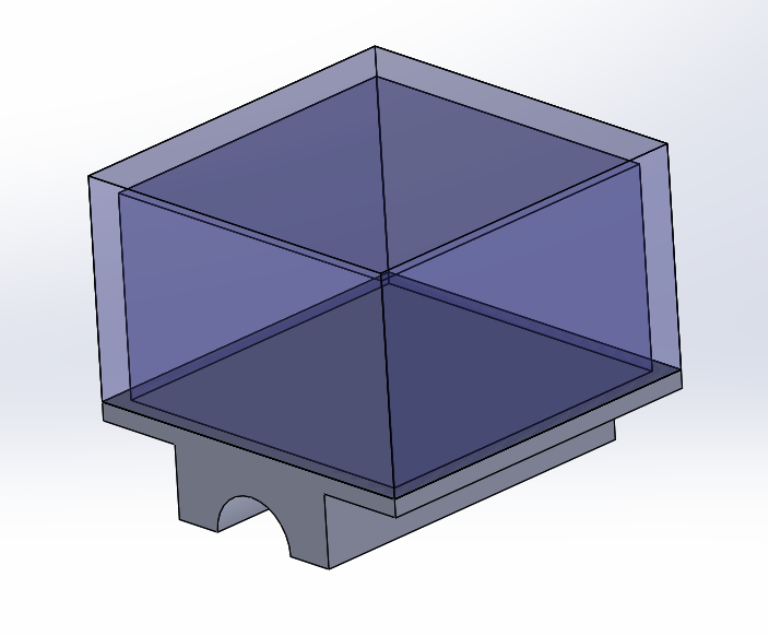


Figure 8.3.10: Preliminary assembly model of the bed and the shell, with the shell made to be transparent for clarity

# 9 Administrative Content

## 9.1 Budget and Financing

The current financing is still in developmental stages. This section will not be able to be accurately completed until the project has been finalized (or is in the final stages). As mentioned in the section on economic constraints, the items currently owned by researchers will be accounted for, but not tallied in the overall price. Also, the financial requirements are understood to be split evenly between researchers.

Table 9.1.1: Budget and Financing

| Item | Quantity | Price |
|---|---|---|
| Smartphone | 1 | $0.00 (owned) |
| Bicycle | 1 | $0.00 (owned) |
| 3D Printed Material | X cubic inches | $15.00 per cubic inch |
| Buttons (with wire) | 5 | $7.99 |
| MCU | 1 | $10.00 |
| PCBs | x layers, x sq. inches | $20 |
| Main Battery | 1 | $50 |
| Alarm | 1 | $< $5 |
| Headlights | 1 | $15 |
| Tail lights | 10 | $10 |
| Turn Signal Lights | 10 | $10 |
| RFID | 1 | $20 |
| Sensor | 2 | $10 |
| Misc. hardware | x | $20 |
| LED Drivers | 4 | $50 |

## 9.2 Milestones

Table 9.2.1: Complete List of All Milestones for the Project

| | Task No. | Task | Date Begin | Date End | Date Due | Prereq. Task(s) | Status |
|---|---|---|---|---|---|---|---|
| Senior Design I (Fall 2021) | 1 | Finalize Group and Project idea | 21-Aug-21 | 1-Sep-21 | | | Completed |
| | 2 | Research and Planning | 16-Sep-21 | 30-Oct-21 | | 1 | Completed |
| | 3 | Work on "Divide and Conquer 1.0" Document | 1-Sep-21 | 16-Sep-21 | 17-Sep-21 | 1 | Completed |
| | 4 | Meet with Senior Design Professor | 20-Sep-21 | 20-Sep-21 | | 3 | Completed |
| | 5 | "Divide and Conquer 2.0" Document | 20-Sep-21 | 30-Sep-21 | 1-Oct-21 | 2,3 | Completed |
| | 6 | 60-page Project Documentation | 1-Oct-21 | 4-Nov-21 | 5-Nov-21 | 5 | Completed |
| | 7 | 100-page Project Documentation | 5-Nov-21 | 18-Nov-21 | 19-Nov-21 | 6 | Completed |
| | 8 | Meet with Senior Design Advisor | 10-Nov-21 | 10-Nov-21 | | 6 | Completed |
| | 9 | Gather Initially Needed Parts | 15-Nov-21 | 30-Nov-21 | | 3 | Completed |
| | 10 | Initial Building and Additional Research (as needed) | 1-Oct-21 | 5-Dec-21 | | 3 | Completed |
| | 11 | Final Project Documentation | 19-Nov-21 | 6-Dec-21 | 7-Dec-21 | 7, 8 | Completed |
| Senior Design II (Spring 2022) | 12 | Layout Physical Specs | 5-Dec-21 | 20-Jan-22 | | 7,8 | Completed |
| | 13 | Build Physical Components (light, alarm, etc.) | 20-Jan-22 | 15-Feb-22 | | 7,8 | Completed |
| | 14 | Build Initial Software Components (app) | 20-Jan-22 | 15-Feb-22 | | 6 | Completed |
| | 15 | Integrate Physical Components with Software | 15-Feb-22 | 15-Mar-22 | | 10,11 | Completed |
| | 16 | Final Testing and Troubleshooting | 15-Mar-22 | 20-Mar-22 | | 12 | Completed |
| | 17 | Finalize Project | 20-Mar-22 | 25-Mar-22 | | 13 | Completed |
| | 18 | Writing Final Report | 15-Feb-22 | 25-Mar-22 | 2-Apr-22 | 14 | Completed |
| | 19 | Writing Website Report | 15-Feb-22 | 25-Mar-22 | 2-Apr-22 | 14 | Completed |

As mentioned briefly in section 4.1, the time constraints are tied to the milestone table above. The beginning and final dates of the design are 'set in stone' (i.e., under no circumstances can they be changed), which can be considered a double-edged sword: the time constraint allows the project to be concluded at a set date without the occurrence of a pushback that designs can be subject to. The other side of this, however, is that the design has to reach a finalized state at the end date, which may cut into extra time for additional troubleshooting and efficiency tweaking. The downside to this immovable due date is probably more of a benefit than a drawback, as seen by the designers; as extra tweaking is an additional step of the design process, and not strictly required for completion of the project.

The following will be a detailed description of the milestone table (see above), in order to clarify any discrepancies that might be observed. The table itself is broken into 8 columns: the first column splits the table into two rows, one for each semester. This can be thought of as an additional date column. As mentioned in section 4.1, the first semester of the project is reserved mostly for design and planning while the second semester is reserved for building and troubleshooting, the milestone table reflects that.

Task No. 1 is to assemble a group and finalize the design idea, obviously this task has been completed; the title page of this report can be considered evidence of that.

Task No. 2 is a preliminary research stage, with group members researching together and separately in order to finalize the core aspects of design (i.e., what will be and/or what should be included and what must be excluded). Technically, the research is still continuing but the preliminary research can be considered completed; the submission of the completion of the summary and introduction page serve as a good indication of that.

Task No. 3 is a specific assignment for the researchers given by the University of Central Florida; it is a document that all group members collaborate on in order to make sure that basic group work ethic is understood. Being an assignment, this task has a set due date (which has already passed).

Task No. 4 involves meeting with a mentor who is in charge of understanding and accepting the design proposal; during this meeting, the design is specified, constraints are mentioned, and general advice is given. Hypothetically, if this meeting were poor, tasks 1 and 2 would need to be re-evaluated.

Task No. 5 is a second iteration of the "divide and conquer" assignment. This is simply an extended look into group ethics and how the members of the group are expected to communicate in the event of argument; this is a service done for researchers by the University in order to avoid (or reduce) group arguments that can deaden projects before completion.

Task No. 6 is the submission of a 60-page project document. This document is meant to act as a precursor for later papers, and is meant to begin a thorough design analysis; this paper requires specification in all aspects of design including constraints, standards, and part specifications.

Task No. 7 is the submission of a 100-page document. This document is an extension of the 60-page document, meant to include more information and include any adjustments that the design mentor notices.

Task No. 8 is to meet again with the design mentor; this is vital as it acts as preparation from the end of the first semester to the beginning of the other. This is the final time that the mentor is seen in person by the group until the construction is underway.

Task No. 9 is the acquisition of parts. This is best done once the design research has been thoroughly done; this ensures that ideally only the necessary parts are purchased, which reduces the price of the design.

Task No.10 is the final task of the first-half, and is when the construction stage 'officially' begins. The parts are gathered and the simple levels of construction are undertaken. Using the information gathered here, the specifications of the 3D printed components can begin to be finalized and even a preliminary print can be made.

Task No. 11 is meant to act as additions to the final paper that will be submitted; the physical specification can be concluded and put into the designers report.

Task No. 12 involves completing the attachments to the microcontroller, which may be done during tasks 10 or 11.

Task No. 13 transitions from the hardware stage to the software stage, and is where the researchers are expected to write the code that will be integrated into the microcontrollers.

Task No. 14 is a more specific step where the code is ensured to run adequately to the designers needs and specifications laid out in the 100 page report.

Task No. 15 is troubleshooting. It is important to place a period of 'breathing room' where design tweaks can be made in case hang-ups occur during any of the previous stages.

Task No. 16 is the finalizing of the project, once the troubleshooting is done and the project can be considered as close to the initial design as possible. Task no. 17 is the writing of the final report, this can be considered the equivalent to the 100-page report done in the first-half. Task no. 18 accompanies task 17, and involves condensing the information in the final report into a website.

The three date columns give the beginning, end, and potential due date of certain tasks. Generally, the end of one task corresponds to the beginning of another, however inspection of the table yields that some tasks have the exact same beginning and end dates. For example, task 12 is understood to be undertaken by electrical engineering researchers while task 13 is to be undertaken by computer science researchers; splitting the work ensures good time management and allows for the researchers to work in their fields of study. Due dates are applied specifically to tasks that are set by the university. It is considered the responsibility of the researchers to set their own due dates in between (which is essentially the milestone table).

The next column specifies which tasks are blocked by prerequisite tasks. These are specified by determining which tasks would absolutely need to be done prior to undertaking the current task.

Finally, the status is a temporary column used by researchers to indicate to themselves and to their mentor that the milestones are being tracked and completed.

# 10 Conclusion

Part 1: Success of Project. Overall this project was a success and performed as intended. It is the same project as designed in Senior Design 1 and boasts only minor design changes as detailed in the following paragraph. While there was some difficulty in garnering success, the team found it to be surmountable.

Part 2: Changes made during design. Changes made during the design include minor things such as the design of the PCB and the removal of consideration of the backup battery implementation and this is reflected in the final design. It was found that the backup battery was not vital and did not fit within our ability to implement within the time constraint. Other minor changes include the removal of an alarm switch on the app this was due to a time constraint and getting the RFID to work without the card. Another minor change made was to the head light controls such that the main control for it is a potentiometer which will set the brightness from a range of off to max as opposed to having a separate toggle and brightness control. The tail light only has a toggle switch as a brightness slider was found to not be necessary. One last notable change is the use of WiFi instead of Bluetooth as there were difficulties getting the code to work properly.

Part 3: Considerations of Improvements. Some possible improvements include using an accelerometer in place of the PIR sensor, adding a GPS for bicycle location, mobile-app based alarm notifications, among others. The design process was effective but would have been smoother if there was a greater familiarity with PCB design

Part 4: Group Rapport. This project is not the most complex but did prove to be a challenge to our understanding of the various topics learned during our respective undergraduate experiences. This group was effective but did struggle with meeting milestones at times though all have been reached.

# References


[1] J. Stromberg, "How and why bicycle deaths happen in the US," *Vox*, May 2014, Accessed: 04-Nov-2021. [Online]. Available: https://www.vox.com/2014/5/22/5738626/how-and-why-bicycle-deaths-happen-in-the-us

[2] "Fatality facts 2019: Bicyclists," *Iihs.org*, Mar 2021, Accessed: 03-Nov-2021. [Online]. Available: https://www.iihs.org/topics/fatality-statistics/detail/bicyclists

[3] "8-bit vs. 32-bit MCU: Choosing the Right Microcontroller for Your PCB Design," Accessed: 31-Oct-2021. [Online]. Available: https://resources.altium.com/p/8-bit-vs-32-bit-mcu-choosing-right-microcontroller-your-pcb-design

[4] B. Giovino, "Take Advantage of a 16-bit Microcontroller's Performance and Low Power," Feb 2019, Accessed: 31-Oct-2021. [Online]. Available: https://www.digikey.com/en/articles/take-advantage-of-16-bit-mcu-performance-and-low-power

[5] "Digikey.com," Accessed: 31-Oct-2021. [Online]. Available: https://www.digikey.com/en/product-highlight/m/microchip-technology/pic24f-16-bit-microcontroller

[6] "Esp32.net," Accessed: 01-Nov-2021. [Online]. Available: http://esp32.net

[7] "ESP32," Accessed: 01-Nov-2021. [Online]. Available: https://www.espressif.com/en/products/socs/esp32

[8] J. Teel, "Introduction to the ESP32 WiFi / Bluetooth Wireless Microcontroller," Accessed: 01-Nov-2021. [Online]. Available: https://predictabledesigns.com/introduction-to-the-esp32-wifi-bluetooth-wireless-microcontroller/

[9] "FreeRTOS - Market leading RTOS (Real Time Operating System) for embedded systems with Internet of Things extensions," Accessed: 01-Nov-2021. [Online]. Available: https://www.freertos.org

[10] "Components  parts / speakers / mono enclosed speaker with plain wires - 3w 4 ohm," Accessed: 03-Nov-2021. [Online]. Available: https://www.adafruit.com/product/4445

[11] "Speaker - 0.5w (8 ohm)," Accessed: 03-Nov-2021. [Online]. Available: https://www.sparkfun.com/products/9151

[12] "Different Types of Batteries for Electronic Products." [Online]. Available: https://www.sofeast.com/resources/different-types-of-batteries-for-electronic-products-guide/

[13] "What type of rechargeable battery to choose for your project?  Battery types and their charging procedure explained," Accessed: 04-Nov-2021. [Online]. Available: https://www.pcbway.com/blog/Activities/What_Type_of_Rechargeable_Battery_to_Choose_For_Your_Project__Battery_Types_and_Their_Charging_Procedure_Explained.html

[14] "Battery Size Chart." [Online]. Available: https://power.tenergy.com/battery-size-chart/

[15] "Which type of battery for smartphone is better: Li-Ion or LiPo ?" [Online]. Available: http://www.benzoenergy.com/blog/post/better-battery-for-smartphone.html

[16] "SMD LEDs," Accessed: 02-Nov-2021. [Online]. Available: https://www.kingbrightusa.com/category.asp?catalog_name=LED&category_name=KCSMD+LED

[17] "LED Chips Explained – differences between 3528s, 5050s and other SMDs," Accessed: 02-Nov-2021. [Online]. Available: https://www.instyleled.co.uk/support/what-are-the-differences-between-types-of-led-chip/

[18] "SMD LED Chips Characteristics Comparison: Size, Power, Efficacy," Aug 2021, Accessed: 02-Nov-2021. [Online]. Available: https://tehnoblog.org/smd-led-chips-characteristics-size-power-efficiency/

[19] C. Rice, "What Are "COB" LEDs and Why Do They Matter?" Dec 2017, Accessed: 03-Nov-2021. [Online]. Available: https://siliconlightworks.com/resoures/what-are-cob-leds

[20] J. Baviskar, "Why are brake lights red ?" Sep 2015, Accessed: 03-Nov-2021. [Online]. Available: https://mechstuff.com/why-are-brake-lights-red/

[21] A. Z. Jones, "What Is the Visible Light Spectrum? Understanding the Colors That Make Up White Light," Feb 2020, Accessed: 03-Nov-2021. [Online]. Available: https://www.thoughtco.com/the-visible-light-spectrum-2699036

[22] "Understanding the IEEE 802.11 Standard for Wireless Networks," *Juniper Networks TechLibrary*, Jan 2018, Accessed: 04-Nov-2021. [Online]. Available: https://www.juniper.net/documentation/en_US/junos-space-apps/network-director3.2/topics/concept/wireless-80211.html

[23] "Understanding Bluetooth Range." [Online]. Available: https://www.bluetooth.com/learn-about-bluetooth/key-attributes/range/

[24] "Beyond Beaconing: Emerging Applications and Challenges of BLE - Scientific Figure on ResearchGate." [Online]. Available: https://www.researchgate.net/figure/Comparison-of-Bluetooth-versions_tbl1_336084528

[25] "MQTT Version 3.1.1." [Online]. Available: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html

[26] "Extensible Messaging and Presence Protocol (XMPP): Core." [Online]. Available: https://xmpp.org/rfcs/rfc6120.html

[27] "HTTP." [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP

[28] "All About Motion Sensors." [Online]. Available: https://www.thomasnet.com/articles/instruments-controls/all-about-motion-sensors/

[29] "Types of motion sensors: Working and their applications," *Watelectronics.com*, Jan 2021, Accessed: 05-Nov-2021. [Online]. Available: https://www.watelectronics.com/types-of-motion-sensors-working-and-applications/

[30] "The Right Tool for the Job: Active and Passive Infrared Sensors." [Online]. Available: https://www.arrow.com/en/research-and-events/articles/understanding-active-and-passive-infrared-sensors

[31] "Know about Passive Infrared Sensor (PIR) with Projects." [Online]. Available: https://www.elprocus.com/passive-infrared-pir-sensor-with-applications/

[32] "Ldt with crimps vibration sensor/switch." [Online]. Available: https://www.te.com/commerce/DocumentDelivery/DDEController?Action=showdoc&DocId=Data+Sheet%7FLDT_with_Crimps%7FA1%7Fpdf%7FEnglish%7FENG_DS_LDT_with_Crimps_A1.pdf%7F11026911-00

[33] S. O'Dea, "Mobile operating systems' market share worldwide from January 2012 to June 2021," Jun 2021, Accessed: 04-Nov-2021. [Online]. Available: https://www.thoughtco.com/the-visible-light-spectrum-2699036

[34] "Core Components and Native Components." [Online]. Available: https://reactnative.dev/docs/intro-react-native-components

[35] "Flutter architectural overview," Accessed: 04-Nov-2021. Licensed under Creative Commons (CC BY 4.0). [Online]. Available: https://docs.flutter.dev/resources/architectural-overview

[36] "Architectural Overview of Cordova." [Online]. Available: https://cordova.apache.org/docs/en/10.x/guide/overview/

[37] "Xamarin." [Online]. Available: https://dotnet.microsoft.com/apps/xamarin

[38] "What is Xamarin?" [Online]. Available: https://docs.microsoft.com/en-us/xamarin/get-started/what-is-xamarin

[39] "Commission delegated directive (eu) 2015/863," *Official Journal of the European Union*, Mar 2015, Accessed: 05-Nov-2021. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32015L0863&from=EN

[40] "BATTERY STANDARDS." [Online]. Available: https://www.epectec.com/batteries/battery-standards.html

[41] "Top 3 Standards for Lithium Battery Safety Testing." [Online]. Available: https://www.metlabs.com/battery/top-3-standards-for-lithium-battery-safety-testing/

[42] "SAFETY CONSIDERATIONS FOR LITHIUM AND LITHIUM-ION BATTERIES." [Online]. Available: https://incompliancemag.com/article/safety-considerations-for-lithium-and-lithium-ion-batteries/

[43] "A Truly Universal Connection," Accessed: 06-December-2021. [Online]. Available: https://www.epo.org/news-events/events/european-inventor/finalists/2013/bhatt/feature.html

[44] "USB Type-C, Power Delivery and Programmable Power Supply," Accessed: 06-December-2021. [Online]. Available: https://www.cui.com/blog/usb-type-c-pd-and-pps

[45] "ISO/IEC 29119," Accessed: 06-December-2021. [Online]. Available: https://en.wikipedia.org/wiki/ISO/IEC_29119

[46] "React Coding Standards and Practices To Level Up Your Code," Accessed: 06-December-2021. [Online]. Available: https://www.jondjones.com/frontend/react/react-tutorials/react-coding-standards-and-practices-to-level-up-your-code/

[47] "ESP32-C3 Series Datasheet." [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-c3_datasheet_en.pdf

[48] "What is RFID? | The Beginner's Guide to How RFID Systems Work." [Online]. Available: https://www.atlasrfidstore.com/rfid-beginners-guide/

[49] "TMS37157." [Online]. Available: https://www.ti.com/product/TMS37157

[50] "TRPGR30ATGA." [Online]. Available: https://www.digikey.com/en/products/detail/texas-instruments/TRPGR30ATGA/4573473

[51] "23-mm Low-Frequency Glass-Encapsulated Transponder, Read Only." [Online]. Available: https://www.ti.com/lit/ds/symlink/trpgr30atga.pdf?HQS=dis-dk-null-digikeymode-dsf-pf-null-wwe&ts=1637258711196

[52] "12V Lithium Ion w/ Built-in BMS." [Online]. Available: https://www.amazon.com/ECI-Power-Rechargeable-2000-5000-Applications/dp/B0973K15QR

[53] "AmazonBasics AA NiMH." [Online]. Available: https://www.amazon.com/AmazonBasics-High-Capacity-Rechargeable-Batteries-Pre-charged/dp/B07NWWHK1J

[54] "Samsung 25R18650." [Online]. Available: https://www.imrbatteries.com/samsung-25r-18650-2500mah-20a-battery/

[55] "Panasonic NCR 18650B." [Online]. Available: https://www.18650batterystore.com/products/panasonic-18650-protected

[56] "High Power LED Lamp Light COB SMD Bulb Chip." [Online]. Available: https://www.ebay.com/itm/224090810254

[57] "High Power LED chip White." [Online]. Available: https://www.ebay.com/itm/262873605755

[58] "Why You Need an LED Heat Sink: Increasing light output  extending your LEDs lifetime." [Online]. Available: https://www.ledsupply.com/blog/why-you-need-an-led-heat-sink/

[59] "High Power LED Aluminum Heatsink Radiator Heat Sink DIY 1W 3W 5W 10W 20W 30W 50W 100W 200W 300W 500W Aquarium Grow Light Bulb." [Online]. Available: https://www.aliexpress.com/item/32952433405.html?spm=a2g0o.store_pc_groupList.8148356.6.2ded670b7bzwKf

[60] "1W 3W 5W 10W 20W 30W 50W 100W Red LED Chip 620-625nm COB Lamp Stage Light Floodlight For 1 3 5 10 30 50 100 Watt Light Beads." [Online]. Available: https://www.aliexpress.com/item/32852299327.html?spm=a2g0o.store_pc_groupList.8148356.6.2b996803mWOqHN

[61] "Multi Color High Power LED Chips 1 3 5 10 20 30 50 100w Watt Royal Blue Green Deep Red Orange Yellow Amber Pink COB Matrix Beads." [Online].

Available: https://www.aliexpress.com/item/1005001329281455.html?spm=a2g0o.store_pc_groupList.8148356.2.2b996803mWOqHN

[62] "10set High Power 1W 3W 5W LED Lens 20MM PMMA Lenses With Bracket 5 8 15 25 30 45 60 90 120 Degree For 1 3 5 Watt Light Beads." [Online]. Available: https://www.aliexpress.com/item/32778141710.html?spm=a2g0o.store_pc_groupList.8148356.16.cc572388jU0DQF

[63] "Twidec/6Pcs SPST Momentary Push Button Switch AC250V/3A AC125V/6A Mini Off(ON) NO Black + Red with Pre-soldered Wires R13-507BKR-X." [Online]. Available: https://www.amazon.com/Twidec-AC250V-Momentary-Pre-soldered-R13-507BKR-X/dp/B08JHSG717/

[64] "Accelerometer Placement – Where and Why," *NXP Smarter World Blog*, Aug 2012, Accessed: 18-Nov-2021. [Online]. Available: https://www.nxp.com/company/blog/accelerometer-placement-where-and-why:BL-ACCELEROMETER-PLACEMENT