University of Central Florida

Department of Electrical & Computer Engineering

EEL 4915L

Senior Design II Group 16

Senior Design Project Documentation

Pill^3

Oriana Alcala : Electrical Engineering

Jordan Schneider : Computer Engineering

Liam Kenney : Electrical Engineering

Fernando A. Oriundo : Electrical Engineering

UNIVERSITY OF CENTRAL FLORIDA

Dr. Samuel Richie and Dr. Lei Wei

Table of Contents

List of Figures

List of Tables

1.0 Executive Summary

This project came into fruition and was designed around the idea that each member would incorporate their own expertise onto the project. The team was made up of three electrical engineers and one computer engineer. Software, hardware and mechanical systems were used to make this project possible. Jordan was a rising computer engineer with a heavy background in coding. Having interned at Amazon this past summer, he is very knowledgeable in his field. With Jordans' background in UX and UI, we utilized his knowledge of software to design the mobile app and interface. Oriana, Liam and Fernando were rising electrical engineers. Their background in analog circuit design, manufacturing and mechanical systems also contributed to the build of the overall project. Together they have interned in a combined total of six engineering companies, helping them establish a good foundation of electrical engineering. While each person had their own set of responsibilities and duties, together they would work on each part as a team to grasp a better understanding of our product and lend help when a team member needs it.

Our overall target of our device was not just the elderly as many would assume, but it was a device that we wanted everyone to incorporate into their daily lives. Aside from prescribed medication, many people across the United States consume daily vitamins and other additional supplements. This involved our beloved pets as well, as many of them also take medication and supplements based on their needs. Forgetfulness doesn't just come with age, it is something everyone deals with. School, work, social events, and stress are a couple of things that we might prioritize in our minds which could lead us to forget. To combat this issue many people face, our product incorporated the need to have multiple subtle notifications. We would have a visual and audible alert that came from the device, but also another that came from our mobile application. Aside from this, another issue many people face was the lack of strength to open the bottle of the vitamin/medication. With our device, the consumer would just need to open it once and fill our container with the product they desire.

Part of the team also has many family members who share these issues themselves. Many loved ones tend to forget to take their prescribed medication. This in result leads to more health related issues leaving many of us worried for their personal health. Due to this, we decided to make a device that would fix this issue. However, inspiration for the device comes not only from our own personal lives, but also from a societal one. Roughly seventy-five percent of all Americans have trouble taking their medication as directed. The lack of self awareness was costing many people in the United States their good health and the health-care industry billions of dollars. Many studies estimate that there are approximately 125,000 deaths per year in the United States that are due to medication nonadherence and that 30% to 70% of medication-related hospital admissions are due to poor management. We intend to solve this growing issue in the United States and better people's lives with our product.

This product was called Pill^3. Pill^3 was an affordable, automated pill dispenser for the masses. While plastic seven day containers and other automated pill dispensers are available in the market, Pill^3 would have a friendly interface that anyone would be capable of controlling. Our mobile app was what makes us standout from the rest. It would set reminders for when you are supposed to take your medication/vitamin, let you set up your schedule from the comfort of your phone, keep track of the times you missed a dosage and also notify you when you are running low on your medication/vitamins. The product would be powered via an AC/DC converter opposed to it being powered by batteries or solar panels. Doing so would allow a steady flow of power without having to remember when to change the battery or having the issue of buying replaceable batteries. Our product was meant to be stationary so the need of batteries to make our product portable isn’t beneficial to us. The snug size and aesthetically pleasing design would also catch the eye of many consumers.

Overall, the Pill^3 was an excellent way to help reduce the issues many Americans face in their daily lives. With its low cost, easy to navigate system and aesthetic design, many Americans across the United States would be able to purchase one. It eliminates the need for organization, and forgetfulness. By also making this device, it eliminates the issue of taking the wrong medication/vitamin, i.e human error. As time passed changes would be made to our device in order to improve its system.

2.0 Project Description

One of the biggest problems in the medical industry was the inconsistency with which patients follow their doctor's instructions. Incorrect dosages, forgetfulness, and substance abuse are among the issues that surround prescription medication and greatly affect the health of millions of patients around the world. These issues are especially prevalent in elderly patients and those with cognitive diseases where it could be difficult for them to remember if they've taken their medicine, or even how much they're supposed to have. This project seeked to create a tool patients could use to accurately track prescription dosage, timetables, and that would dispense the correct amount of medication according to the information given.

Incorrect or complete lack of consumption of prescription medicine was one of the leading causes of adverse clinical outcomes. In the United States, non-adherence rates for chronic conditions like diabetes are as high as 50%, and could cause around 125,000 deaths per year. To help alleviate these issues, there are many devices in the market available to help patients keep track of prescriptions, such as plastic pill containers that are separate by weekday, or automatic pill dispensers that are refilled via pharmacy delivered trays. These devices, while helpful, still have issues, primarily of which was the price. The two readily available market models revolve around monthly subscription payments for access to an app or a one time purchase of a device.

To solve these issues, this device would be affordably produced, not requiring expensive components or costly server maintenance. This device would strive to deliver a streamlined setup and refill experience that would make it both simple to maintain and easy for patients or caregivers to operate. A smart refill compartment would make adding new medication a breeze, and a built in tracker would alert patients when a prescription was running low. The sleek design helped minimize fault points making it reliable, and the app interface provided any relevant information about the current prescription at the click of a button.

2.1 Project Motivation and Goals

The motivation behind this project was to help those who tend to forget to take their medication. It could be something as simple as not taking out vitamins which might not have a big impact on our health, however, there are people who need to take a specific medication at a specific time and day to avoid any kind of diseases or to even control a health condition, hence, taking that pill on time could be a difference between life and death. Reports say that " Seventy-five percent of Americans have trouble taking their medicine as directed [1]" This lack of adherence was costing many people their good health and the health-care system billions of dollars. Estimates are that approximately 125,000 deaths per year in the United States are due to medication nonadherence and that 33% to 69% of medication-related hospital admissions are due to poor adherence [2]. The total cost estimated for medication nonadherence range from $100 billion to $300 billion every year, when both direct and indirect costs are included.4–8 Also, nearly half of all Americans—133 million people—suffer from at least one ongoing or chronic health condition. That number was expected to grow to 157 million by 2020 [3]"

These numbers showed how important it was to adhere to the times and days that are prescribed by the doctor to take these medications. Most of us also have our family members that tend to forget to take their medicines and we are the ones constantly reminding them to take them at their respective time. Each of the developers of the Pill^3 had different situations in which their loved ones had been affected by this kind of problem. Starting with Oriana Alcala, she had been given the duty to make sure her grandfather took his medication, although her grandmother helped, there were times in which she was unable to remind her grandfather to take the prescribed pills. Since she has such a busy schedule there are times that she has to call last minute to confirm her grandfather has taken his medication. This became an endless cycle for her which led to stress and headaches.

For Fernando Oriundo, the case was slightly different. Fernando struggles everyday to not to forget to give his dying dog the prescribed medication. Fernando was under so much stress since one missed pill could cause a big impact on his pet health. He tried to set numerous alarms on his phone in order to not forget but at times with school and work he would have to ask his parents or siblings to give his dog the medication, as well as Oriana Alcala, he was under a lot of stress trying to make sure his dog took the pills at the respective time.

For Jordan Schneider, it was as simple as forgetting the vitamin supplements. He would carry the classic pill container that could be purchased at any store. He would have to set up alarms in order to remember to take these supplements since he was a person who constantly works out. Similarly, to the other members, he was someone who is constantly busy and at times tends to forget to take these medications which affected his health and goals.

Lastly Liam Kenney was in the same situation. The difference was that he was very familiar with the medicine field due to the influences from his parents. In fact, one of the reasons we decided to build the Pill^3 was because of these daily problems each of us faced. Liam had a great source of information which we would use to see how so many people would be affected by simply missing their medications.

Because of situations like these, we decided to create the Pill^3 that would help our loved ones and even ourselves. As we age, our body demands certain supplements that we would need to be taken constantly. This product would not only target the elderly consumer but also the young population. The idea was that the product could help anyone who needed to take any medication on time.

The idea behind the Pill^3 was that it would be an automated pill dispenser that through an app it would alert the consumer when it was time to take certain medication. Not only that but it would also alert them when they missed medication on the day that was scheduled. Another feature we would have in the app in terms of the alert system was that it would let the user know when they are running low on specific medication, instead of waiting when the container was empty. The user would be able to access our app from any other device if they wanted someone else to get the remainders. Lastly, they would be able to set up their pill schedule into the app and select what day that medication should be taken, so that when the container was empty it would let the user know that there was a spot available to add more rather than needing to be refilled.

In terms of the mechanical components, the product would dispense the pills in a small cup for the user to just take. There would be inner compartments where the user would place all medicine needed, the design so far does not have much in terms of the equipment itself since it would be better if the user interacted with the product through the app rather than the physical equipment.

Lastly, our goal would be making the actual product affordable, we analyze, and research similar equipment's and they are very expensive, we want to make sure that the users could buy it at a reasonable price, and they are not hidden fees for the app or any other additional features, since our goal was to reduce the number of casualties caused by the forgetfulness behind taking their medicine.

Based on the early stages of the project and the design we have established the main things the device should do in terms of the app and the device alone. This would be to satisfy our targeted audience and potentially be a competitor for the existing products in the market. With the research made we have decided to have the following in the device:

The device should:

- Be able to get the signal from the app and display it in both phone screen and the actual device screen
- Be able to send the user the following notifications:
    - When the container was empty
    - When there was a medicine to be refilled
    - When the medicine has been dispensed
    - When the user forgets to take their respective medicine
- Be able to receive and store information given by the user about scheduled times and expired dates for each of the prescriptions
- Be able to identify the pill that was programmed to be dispensed
- Be able to sensor when the medication was dispensed
- Be able to know based on weight using a timer if the medicine was taken by the user if not notify the user of the missed pill
- Be able to determine when the cup was placed in dispensing area
- Be able to not let any other unauthorized user to access to the pills

2.2 Marketing Requirements

The Pill^3 was a device meant for people of all ages. While it was true that a majority of our demographic was the elderly, we've incorporated and emphasized the fact that our device was meant to be used for vitamins as well. This would resonate with the youth since this generation was more health conscious than its predecessors. In order to ensure our product reaches the masses, it would also be beneficial to partner with clinics, doctors and retail stores. By doing so, patients would be more informed since these are the only sources where medications and vitamins could be purchased. Unlike our competitors, the Pill^3 would also be purchasable at a lower cost and offer similar, if not greater features.

2.2.1 Social Media

Alongside traditional television advertisements, the use of social media has been growing over the past couple of years and would continue to grow as technology progresses. Linkedin, tiktok, Facebook, instagram, youtube, twitter and snapchat are some of the biggest platforms out at the moment. Utilizing these forms of social media would help spread publicity for Pill^3 reaching the masses.

A simple way to inform the general public about the Pill^3 on social media was by having ads play and/or run in these social media sites. As shown in the table below, each platform varies in demographic. By taking part in these platforms and showing ads, we could further our reach and inform a greater number of people.

Table 1: Demographics

Social Media	Demographics
Linkedin	Over 60% of users are over 36 years old
TikTok	Over 60% of users are less than 30 years old
Facebook	Over 40% of users are 25-35 years old
Instagram	Over 60% of users are 18-34 years old
Youtube	Over 70% of users are 15-25 years old
Twitter	Over 40% of users are 25-34 years old
Snapchat	Over 50% of users are 18-34 years old

Aside from paying to have our ads play on these platforms, we are also able to create accounts and interact with the customer. A perfect example of social media usage was Wendys. In 2017 Wendys use of banter with its customers and sass on social media had a huge impact on its sales. Already being a multimillion dollar company, they were able to have a 49.7% increase in growth. Their overall profits increased from 126 million to 194 million dollars for the year. Globally, Wendys was able to exceed 10 billion in sales. Following this year, many other companies started to implement this onto their own social media platforms and many have seen success. If we follow the trend, we should also be able to benefit from this tactic.

2.2.2 Outreach

Another benefit of using social media and having a mobile app was the ability to obtain feedback. While we would test the hardware, software and the mechanical aspects of our device, having constant feedback from an audience would be beneficial to everyone. Certain defaults or accidents that may have happened that we unaccounted for could occur. By bringing the issue to light, we would be able to act on it quickly and incorporate a solution to the issue at hand. Satisfying every party.

Some ways that we may be able to do this was by the use of our mobile app. We could set biweekly reminders to the user to give us feedback on how the application works in their possession. Glitches and in-app errors could vary based on the user's phone. This would also allow us to constantly keep updating the application so that errors may be erased. Aside from this, the user could suggest features we might have overlooked, providing for a more enjoyable in-app experience.

Sending mass emails to the user who obtained the Pill^3 could also be very beneficial to us. Many people tend to forget to check their application after completing the setup process. Since one of the main applications for our device was to be a reminder, checking the application would not happen as often since this notification would appear on the home screen of the user's phone. The purpose of the emails would be to notify the user of new updates and to provide us more feedback on the device. These types of feedback would also be tailored more towards the components opposed to the phone application.

Having both of these types of feedback, we would be able to improve our device in every aspect. Regardless if it was a software, hardware or mechanical issue.

2.3 Objectives

The main goal of this project was to create a device that assists a wider variety of users with taking daily supplements or prescriptions. To achieve this, the device would be sized so that it could fit on the typical kitchen or bathroom counter and obtain power from any home outlet through a wired connection. It would contain a speaker for audio cues, a screen for visual information about the prescription or supplemental regiment, and a wifi connection to a smart phone app that would send notifications about the status of pill storage or an upcoming dosage.

The device would be able to keep time and monitor pill levels so that the user would need to do nothing more than an initial setup and calibration for each prescription. To log time the device would use a microcontroller crystal clock, as power consumption was not an issue and a crystal would be more accurate than an RC network clock. To check pill levels, the device would monitor the initial weight of the pills and using the user entered amount of pills, calculate how much each pill weighs. With this, the device would be able to report to the app the current number of pills for a prescription left.

To complement the device, a smart phone application would allow the user to input information about each prescription including the number of pills, how many pills are supposed to be taken at each interval, and how often to schedule each interval. To accomplish this, the application would connect to the device through a wifi network and, once paired, would default to that device while on that network. The application would offer a calendar to view and schedule prescriptions and a status screen for each individual medicine that offers a variety of information. The application would also send push notifications to the user based on the settings they selected, alerting them to dosages or refill reminders.

2.3.1 Goals

Our team had several additional goals that we planned to achieve in order to meet all of our objectives. These goals were either extra features we wished to add, constraints we wanted to adhere to, or design ideas we wanted to fulfill. Each goal was discussed briefly below, and the reasoning for its inclusion was listed.

Price Point: Most single purchase competitors with similar features market their products for significantly more than what this device was planned to cost. Other competitors offset the cost with a monthly subscription model that this device would not use. As such, this device seeks to be the one of the most advanced for its price class. This may cause issues with components needed for specific features. This device would seek to avoid these issues by smart use of less expensive components while still maintaining quality and reliability

Size: This device would ideally be large enough to hold a sufficient amount of medication while not being cumbersome to move or fit into a home setting. The size limit of 1.5 cubic feet would allow, with good use of internal space, for the device to meet the capacity requirements while not taking up excess counter space. This may cause other issues to arise and limit the types of components that could be used internally and externally. To solve one of these issues, no internal power supply would be added, with the device instead being powered through a cord.

Interactivity: This device was designed to be connected to a home wifi network and for prescriptions to be entered through a companion smart phone application. As such, it would not operate in environments where wifi is not available. While this was an issue, most environments where this device may be set up are likely to have both a wifi connection and a smartphone device with which to interface.

Types of Prescription: One issue of dispensing prescriptions or other supplemental pills was the large variety in shapes and sizes that are commercially available. This makes it difficult to create a “one size fits all” measuring and distribution device. In regards to measurements, the device would weigh the total amount of pills, and divide it by the user entered number of pills, thus obtaining an estimate of the weight per pill. This weight could then be used to monitor medication levels and distribution weights.

Technical Error: An issue that may arise would involve the device losing power and having to restart causing it to lose all of the information stored. In order to combat this difficulty, the app on the user's smartphone would need to store the information that was inserted and sync it with the device in case the user encounters this kind of situation.

Child Safety Lock : Majority of these kinds of products deal with safety issues due to the nature of accessing these medications easily. Ideally the device could have a passcode or lock to avoid kids or any unauthorized user to access the patient medication.

2.4 Requirements Specifications

The initial requirements for this project were discussed and decided upon during our team's first meeting after being approved to continue with the design of the Pill^3. Both rudimentary hardware and software requirements were chosen to better shape the schematic and purpose of our project. This section briefly lists these specifications and provides a small explanation for the purpose of each one.

2.4.1 Hardware

- The device occupies a space within 1 cubic foot
    - Our team wanted to make a device that was both large enough to store at least 1 months worth of three different types of medicine and still be small enough to fit on the average consumers kitchen counter
- The device would cost less than $200 to produce
    - During competitor research, our team saw that most competitors either had a large initial price for the device, or a rather hefty continual payment plan. We decided that we wanted our device to be an affordable option for consumers
- The device would contain 30 days of medication with pills sized at 14 mm
    - In order to make this device actually useful to consumers, our team decided that the minimum number of days of medicine it should hold should be one month, as typical prescriptions tend to be dated for about that long
- The device would be powered via USB cable
    - This device would take AC power and turn it into DC power using a converter, but our team decided that using a USB cable to connect the Pill^3 to an adapter in the wall outlet would allow for more flexibility for the user
- The device would contain a microcontroller
    - Used to manage in device systems
        - Time
        - Notifications
        - Mechanical interfaces
- The device would contain a speaker capable of producing 8 kHz sounds for audio cues
    - 8 kHz was well within the human range of hearing, so our team decided that this would be a good level to have audio alerts. These alerts would let the user know when it was time to take their medicine if they are near the device
- The device would contain a 4” display for visual information and cues
    - This screen would allow the user to check on the information of the system without having to open the smartphone application, making the device easier to use
- The device would contain 3 IR sensors, one for each prescription
    - Our team decided that these sensors would be the best type in order to accurately track medicine dispersal and amounts within the storage device
- The device would contain an apparatus to dispense medicine into a cup
    - While the exact design has not been decided on, this device would need some sort of mechanism to get a single pill from each of the storage containers and be able to dispense it to the user

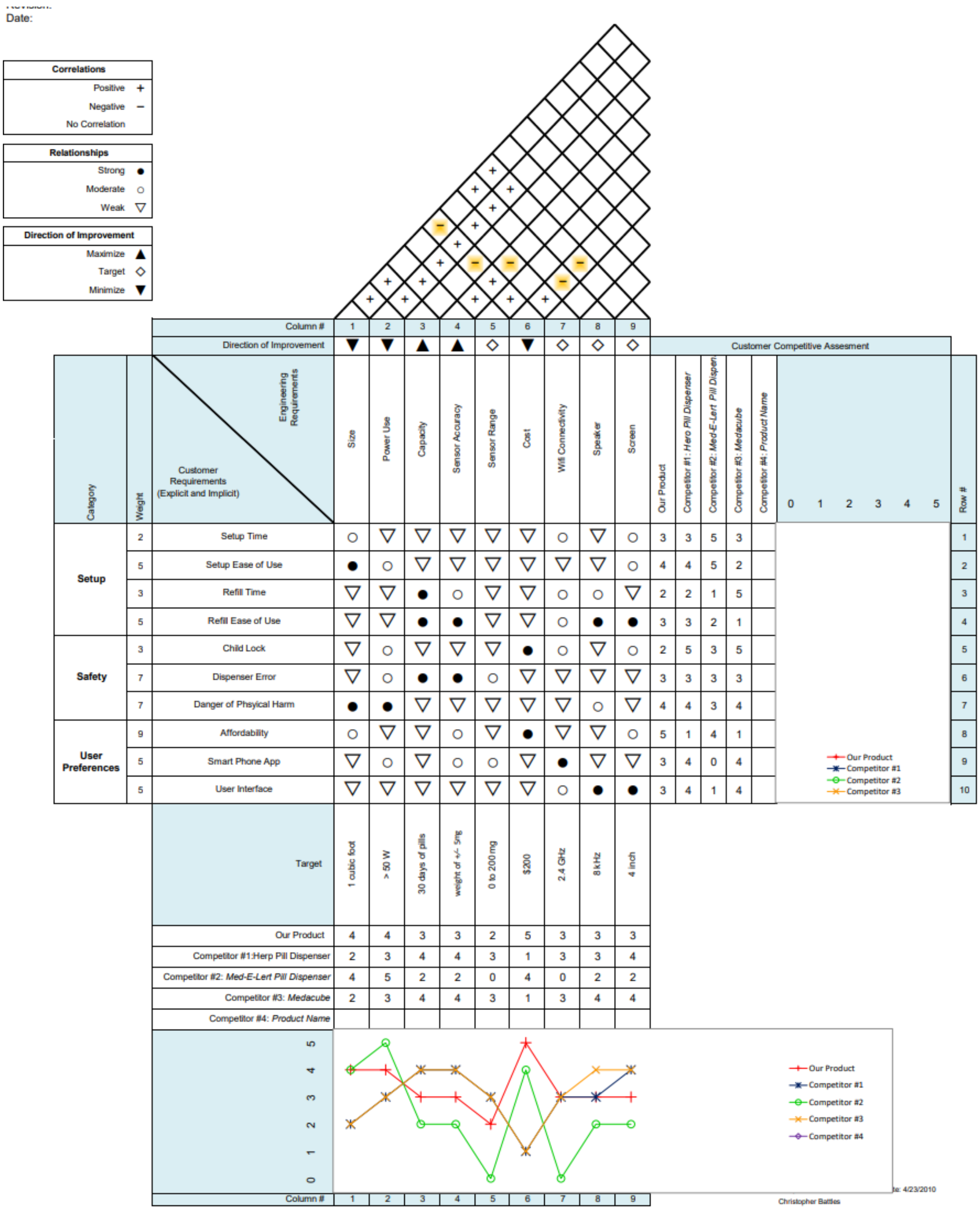
- The device would have a pressure sensor that could tell there was a cup to dispense the medicine
    - In order to avoid spilling the medicine accidentally, our team decided that this device should have a small sensor underneath the cup where the medicine was dispensed to, to ensure that the cup was there before anything was dispensed

2.4.2 Software

- The device would connect to a phone app via 2.4 GHz wifi
    - In order to make the app easy to use for the consumer, and accessible to most, our team chose this band of wifi to transmit data from the device
- The app would send push notifications when medication is dispensed
    - This type of notification system would allow our device to alert the user from their smartphone anytime there was medicine dispensed
- The app would send push notifications when prescription is 2 weeks away from running out
    - This type of notification system would allow our device to alert the user from their smartphone anytime there was a low supply of medicine
- The app would send push notifications once it was time for the user to take the medication
    - This type of notification system would allow our device to alert the user from their smartphone that they forgot to take their medicine at the prescribed time
- The app would display data about prescription
    - Remaining amount by
        - Weight
        - Approximate days left
        - Number of pills
    - Dosage
        - Amount taken per day
        - Time taken
        - Number of pills per dose

2.5 House of QualityAnalysis

Shown below is the House of Quality, a graphical representation of the priorities the team has ascribed to certain aspects of the Pill^3 and how our device compares to industry competitors. Customer requirements and engineering requirements are ranked on how strongly they relate to each other within the Pill^3's proposed design. The shapes used denote whether a certain customer requirement would change the amount of focus or research needed in an engineer requirement. The pyramidal “roof” of the graph was used to show the correlation, whether positive or negative, of the different engineering requirements. This helped the team to analyze how different parts of the project related to each other, and how likely a change to one component would affect another. The right side of the graph shows the weight given to each consumer requirement, which acts as a list of importance decided on by the team. The right side of the graph was a comparison of how the Pill^3 performs in each of the consumer requirements and how each of the competitors does. The bottom of the graph visualizes the performance of each product in a format that was easy to see.



Category	Weight	Column #	1	2	3	4	5	6	7	8	9						Row #
		Direction of Improvement	▼	▼	▲	▲	◇	▼	◇	◇	◇						
		Engineering Requirements / Customer Requirements (Explicit and Implicit)	Size	Power Use	Capacity	Sensor Accuracy	Sensor Range	Cost	Wifi Connectivity	Speaker	Screen	Our Product	Competitor #1: *Hero Pill Dispenser*	Competitor #2: *Med-E-Lert Pill Dispen*	Competitor #3: *Medacube*	Competitor #4: *Product Name*	
Setup	2	Setup Time	○	▽	▽	▽	▽	▽	○	▽	○	3	3	5	3		1
	5	Setup Ease of Use	●	○	▽	▽	▽	▽	▽	▽	○	4	4	5	2		2
	3	Refill Time	▽	▽	●	○	▽	▽	○	○	▽	2	2	1	5		3
	5	Refill Ease of Use	▽	▽	●	●	▽	▽	○	●	●	3	3	2	1		4
Safety	3	Child Lock	▽	○	▽	▽	▽	●	○	▽	○	2	5	3	5		5
	7	Dispenser Error	▽	○	●	●	○	▽	▽	▽	▽	3	3	3	3		6
	7	Danger of Phsyical Harm	●	●	▽	▽	▽	▽	▽	○	▽	4	4	3	4		7
User Preferences	9	Affordability	○	▽	▽	○	▽	●	▽	▽	○	5	1	4	1		8
	5	Smart Phone App	▽	○	▽	○	○	▽	●	▽	▽	3	4	0	4		9
	5	User Interface	▽	▽	▽	▽	▽	▽	○	●	●	3	4	1	4		10

	1	2	3	4	5	6	7	8	9
Target	1 cubic foot	> 50 W	30 days of pills	weight of +/- 5mg	0 to 200 mg	$200	2.4 GHz	8 kHz	4 inch
Our Product	4	4	3	3	2	5	3	3	3
Competitor #1:Herp Pill Dispenser	2	3	4	4	3	1	3	3	4
Competitor #2: *Med-E-Lert Pill Dispenser*	4	5	2	2	0	4	0	2	2
Competitor #3: *Medacube*	2	3	4	4	3	1	3	4	4
Competitor #4: *Product Name*									

Figure 1 : House of Quality

3.0 Existing Similar Projects and Products

Although some of the existing products have a lot of functionality and are very efficient, there are a couple of things to take into consideration, the first one being the cost, the price ranges from \$50 - \$1800 which not many people are able to afford. One of our goals was to create a product that was affordable to the masses and that could potentially have all the features that some of these products offer.

Another problem consumers faced was that these products are not user friendly. Not many people are comfortable with modern technology, and in fact there are some consumers that struggle using a phone or computer "According to the results, 80% of the older people had regular mobile phones and 20% had smartphones. In 95% of the male and 80% of the female participants, the greatest use of mobile phones pertained to making phone calls. A total of 5% of the male and 2% of the female participants used the Internet on their mobile phones. A total of 44% of the female and 42.80% of the male participants had poor attitudes (score from 0 to 40) toward mobile phone use. [2]."

Based on the product specification and usage, we noticed that they must follow a specific way to use the product to either get the user pills dispensed or just have them refilled. The whole purpose of the product was to help the user to take their pills on time without causing headaches. That was when the Pill^3 came in, we would create an interface for the consumer that would be rather simple to navigate, giving them the ability to use the product without any problems.

Surprisingly there are quite a few competitors in the automated pill dispenser market, some of these existing products have either a lot of functions or not many, each of them target a different audience and some of these do not have an app for it making the product too simple for the price paid. Some of the products that are similar to ours are: Hero Pill Dispenser, Med-E-LertTM Premium Locking Automatic Pill Dispenser, MedMinder, the Livi, and Pria.

3.1 Hero Pill Dispenser

Our first competitor was the Hero Pill Dispenser. Hero's business plan was an initiation fee with monthly payments, which provide a way to turn down consumers. Having a lower upfront with monthly payments of \$30 could steer away customers who don't want what Hero considers an "assisted living" device. There are plenty of consumers who don't want this as a necessity, but also want to be able to use it for pills such as vitamins, supplements, but also prescription medicine.

Aside from this fee, there was also the cost of the equipment itself which could go as high as \$100. Not only that, but the Hero Pill Dispenser targets mostly those who have prescribed pills since the containers for them are not big enough to hold a vitamin pill which in some cases tends to have a large size. With Pill^3 the user won't face that problem since the product was designed to dispense any kind of pill without having any size constraint. In summary the Hero Pill Dispenser might be "affordable" in terms of the equipment, however, there are some added fees that make the product more expensive. The Pill^3 would only require an upfront cost to cover the price of the device. There was no need to have a subscription service as people don’t always want this as an assisted living device

3.2 Med-E-Lert™ Premium Locking Automatic Pill Dispenser

Next, we have the Med-E-Lert™ Premium Locking Automatic Pill Dispenser**,** the Med-E-Lert works on a daily or on intervals throughout the day. This, although it would be considering a competitor, was a slightly different design than what we plan to make, their design was on the low end compared to the Pill^3 since all the products really do was to set the alarm per day for the pills the customer organized and it does not dispense it, you must shake or tilt the apparatus to be able to get the pills. On the other hand, our design would require a schedule set on an app that would independently dispense pills. A downside of the Med-E-Lert was that this design was not customizable and requires you to manually fill each day when you run out, not only that but the size of the slot for the pills was small and could only a specific number of pills that the again would fall in the size constraint mentioned before. In addition to these constraints, this product was designed for only every 28 days, if the user must take pills twice a day this cuts the length of the stored pill, this would last down to 14 days as you must take pills more often throughout the day it cuts down how long each cycle would last.

Another downside to their design was that there was no way to notify the user without them currently being in the room. If someone was out of their home, this would be ringing constantly until someone was able to come back and turn it off. In the case of this device, it would ring, no one would hear it, and then the user would never actually go back to check to take their pills for the specified time. This was counter-intuitive and would result in someone not taking their daily pills hence why the app would be extremely useful for the user since the phone was one device that we are constantly carrying. Then again, although this product does not follow the same idea of dispensing the pills it was still considered a competitor since the price of it was not as bad as other devices, nevertheless, for what the product really offers, the price of it was too high.

3.3 MedMinder

Next, we have the MedMinder, this product was interesting on the design side. The product does not have the dispensing option, instead it has compartments that the user has to open to access the pills. These compartments would light up to let the user know which pill slot they have to open to avoid opening the wrong pill slot. This concept, while interesting, was rather inefficient since someone must refill each of the slots, the amount available was 28 spaces only which would be ideally for 28 days, but it would really depend on the frequency the user has to take the medication. One of the good things about the product compared to other competitors was that the space to store the medication was slightly bigger than other products, however, Pill^3 would offer the customer a larger space to store more pills and not refill the containers too often.

The MedMinder also has an interactive screen for the user, this screen would display information about the patient, and it even has an option to show pictures and videos. While this was innovative it's quite useless, the device alone main purpose should be to alert the user when the pills need to be taken as well as when one of the compartments are empty. MedMinder was providing the user a pill storage and an electronic picture frame, which you wouldn't be placing your medicine storage in the living room for everyone to see, which could potentially be a major problem since a lot of people would be able to take the medication as well as have easy access to it.

One downside to their design was that they needed a bracelet to alert the patient when they must take the medicine; that was the alert system they implemented. It may appear as a good strategy but at the same time was not, if the user were to lose this bracelet, then they could no longer receive any notification when it was time to take the respective medication. On top of that, the bracelet itself was not waterproof, although this was a minor thing, it should still be consider since the goal of the product was to let the user know when it's time to take their prescription, if the bracelet were to get wet and hence damaged the user would not get this notification, on top of that they would need a replacement for it which would cost extra money to the user to get it replaced.

With Pill^3 that won't be a problem since our interface could work with mobile devices " … 28 percent of consumers would drop their phones into liquid and 8 percent spill onto their device Water damage are the two dreaded words that are not covered by the one-year manufacturer's warranty." The unlikability of dropping the phone in water was very low, taking this factor in consideration it would make our product more efficient since the user could access the app through any phone with their username and password.

3.4 Pria

Another big competitor was Pria**.** This was an automated pill dispenser; it followed the idea of the Med-E-Lert and the Hero pill dispenser. It has a very user-friendly interface, however, it's limited in terms of space. The Pria connects to the user's phone giving them notification when it's time to take the pills and it has a couple of other features that are not relevant to the pill dispenser. It has the limitation of storage. Each compartment was slightly bigger compared to other products, but it could only store up to 28 doses, which for some users might be enough, but others may not suffice, and the idea was that the product helps any user regardless of age.

Overall, the Pria does deliver what was expected, but it has its limitations. The Pill^3 would not have that problem since the design was for all ages and circumstances. There might be people who need to take certain medications even three times a day, we would have that in consideration while building our products and this was to ensure we have a broader consumer group. Similarly, to the Pria, the Pill^3 would have an app that notifies the customer when it's time to take their medication as well when the equipment needs to be refilled.

3.5 The Livi

Then we have The Livi pill dispenser, this product also follows the concept that Pill^3 wants to achieve, in terms of equipment it's quite like ours. The Livi does not give the user the option to have the medicine automatically dispensed, instead you must press a button to get the medication dispensed. Similarly, to the Med-E-Lert the user needs to be around the product to be notified when it's time to take the medication. This could be a problem since then again, if the user were to forget to take the medication and they are not around the product, then there was no way for them to take it at the respective time. The notification system for this product could be considered rather inefficient.

In other words, The Livi provides a lot of features that are useful to the user, and it was not as costly as other products, however, this product only targets an older consumer which limits the scope of the product. The design of it it's also quite complicated since it has some buttons that are not defined in the product itself making it very likely for the consumer to not know and must rely on the manual or on someone to help the consumer to use as it was also shown on the website for this specified product. The idea was that the product alone was easy to use, and the user could navigate through the app if they were to face any problem.

3.6 Medacube

Last, we have the Medacube**.** This product has a great design, was easy to use and the screen implemented gives the user a lot of information about the pills alone and the pills that would be dispensed throughout the day. Added to this it has videos in which it explains how to use the equipment efficiently and, in the video, they explain everything step by step. Their notification system, and schedule setting interface and all with a lack of a subscription service.

However, their product was about $1800 which was extremely overpriced for what the device does. Compared to the other devices this product was more efficient but in terms of cost not every user was able to get it, although it targets a wide range of users it was hard to attain. The Pill^3 plans to make a product very similar to Medacube, but instead a cheaper alternative. They do not have an app, so setting your schedule was a little more difficult as you must be present at the device. It was the same with all the configurations. We would be able to provide an easier way to set the schedule for the patient and have a better notification system which allows the user to know they must take the notification and they could keep track whether they took the pills or not.

4.0 Relevant Technologies

We would showcase the relevant technologies that would be used to make the Pill^3 functional. We would discuss each item, how it works, compare their features and later in the section we would bring to light which ones we choose for our device. It was important to know about the different types of devices as it could greatly improve our device.

4.1 Sensors

Sensors are devices that could detect a change in its environment. Unlike an actuator, sensors convert a physical change into a measurable electrical signal. They are widely used in many modern day devices. Sensors, also known as transducers, could also fall into an abundance of categories. Depending on the usage, a sensor could be classified as active, passive, biological, photoelectric, thermoelectric, analog and digital. For simplicity, we would only be focusing on analog, digital, passive and active sensors.

Analog sensors are sensors that produce an analog output. These sensors are used to measure external factors such as solar, wind, sound and more. The output of an analog sensor could range from 0 V - 5 V. When logic was high or “1”, the range was from 3.5 V - 5 V and when logic was low or “0”, the range was from 0 V - 3.5 V. Some examples of analog sensors would be pressure sensors, sounds sensors, light sensors and accelerometers.

Digital sensors are sensors that produce a digital output. Digital sensors are measured using discrete values, meaning that the data was either high or low. It was here that data was converted into digital form and transmitted. Most digital sensors are within the limit of 0V - 3.3 V. Some examples of what a digital sensor might measure would be conductivity, pH values and temperature.

Passive sensors are sensors that measure natural emission. These types of sensors require a source of illumination. Passive sensors measure the data of vibrations, light, radiation, heat and other environmental factors. Active sensors are sensors that provide their own energy. In order to power an active sensor, an external source was needed. These sensors are used to measure signal transmission.

For our Pill^3 device a variety of sensors would be utilized to have a more accurate and efficient way to have the pills dispensed for the customer. For this design, research has been done to better determine which sensor would work best with the other components that would be incorporated into the device. The main goal of these sensors was to determine when a medication has been dispensed and when the medication has left the storage area; other applications would be taken into consideration during the design. The advantage of these sensors was that most of them do not require a lot of power to work, in fact they could work with as little as 3.3 Volts, as for the size, it would vary per sensor, although these sensors are not large in size we would take into consideration how big each sensor was since the smaller the sensor the better would be for the device, since we could save some space that might be needed for larger components. To determine which sensor would be utilized, we analyzed different applications as well as constraints that each of these sensors could bring.

4.1.1 IR Sensors

An IR sensor (infrared sensor) is a radiation-sensitive optoelectronic component that has a spectral sensitivity in the infrared wavelength range of 780nm-50um. These sensors are commonly used in alarm systems to determine when there are unauthorized users in the area or even to switch lamps on or off. For the Pill^3 device these sensors would be implemented to determine when a pill has been dispensed. Each of the storage spaces would have a separate sensor that would communicate with the MCU unit; the idea was that the MCU unit would keep a count on the number of pills left in the container. An advantage of these sensors was that they are relatively small, do not require much power, and are able to work with a short range between the sensor and the object. For the Pill^3 purposes this works perfectly, the only disadvantage of these sensors was that if we were to move the sensor further away from the target it might not be able to detect it, these sensors are sensitive to hard objects like a wall and even dust, nevertheless, for the Pill^3 it meets all the needed criteria to detect small objects such as pills.

4.1.2 Ultrasonic Sensors

An ultrasonic sensor is an instrument that could measure the distance of an object using ultrasonic sounds waves, this sensor uses a transducer to send and receive ultrasonic pulses that are based on the object's proximity. When researching for sensors, the ultrasonic sensor was considered since we have been exposed to them in different applications from our previous classes. The Ultrasonic sensor has more disadvantages than advantages for this project. The main disadvantage of the ultrasonic sensor was that it has a slower reaction rate compared to other sensors, for the Pill^3 the sensor that would be utilized needs to be able to detect the pill immediately as it is falling since it would send an automatic signal to the MCU unit in relation to the amount of pills left in the specific container, not only that but it has to send a signal when a pill has been dispensed. Because of this, the Ultrasonic sensor would not be a good choice for the Pill^3. Added to this, the ultrasonic sensor was larger compared to other sensors, even though it might seem small, it could make our overall design larger than expected. In conclusion the Ultrasonic sensor would not be able to meet our needs.

4.1.3 Temperature Sensors

A temperature sensor is an electronic device that measures the temperature of its environment, based on this could convert the input data into electronic data to record, monitor, or signal any changes in temperature. The Pill^3 won't be needing this sensor for the pill dispensing process, however, based on the motor we decide to utilize for our product a temperature sensor could be useful to activate a cooling system (fan) next to the motor to avoid any sort of overheating. The number of sensors needed would be just one since then again, the only usage for this sensor would be the actual motor. The idea was that this sensor could communicate with the MCU and activate the fan when it detects high temperatures, and we could avoid any potential internal damage of the device.

4.1.4 Weight Sensor

A weight sensor is a type of transducer, specifically a weight transducer. It was able to convert an input mechanical force such as load, weight, tension, compression, or pressure into another physical variable, in this case, into an electrical output signal that could be measured, converted, and standardized. The Pill^3 would be using this sensor to detect the cup that needs to be placed in the dispensing area; this cup would be a kind of "signal" to the MCU unit that simply states that the device is ready to dispense any scheduled pill. This sensor would send a signal to the MCU unit letting it know that the cup was in its designated area, once this signal has been sent, the Pill^3 would be ready to dispense the pills that were scheduled to be dispensed that day. The idea of this sensor was to be also linked with the notification system since it could alert the user that the pills has been dispensed, not only that but after an established time it could determine that the user did not take the pills based on weight and that the cup need to be fully empty before it could dispense any other medication, it could keep track on the app that the user missed a dose.

4.2 Power Supply

The power supply was the heart of the device, it is what drives each component. For the Pill^3, we took into consideration several different types of power supplies that could be used for our design that met our requirements. When deciding on what type of power supply to use, we took into account many different factors. Portability, efficiency, cost and functionality are what we based our decision on. The power supply would need to be able to stream a constant source of power, fit in the casing and be convenient for the consumer to use. We would be looking at several power supplies to determine what is best for our device.

4.2.1 Battery

Batteries are one of the most common forms of power on the market. It comes in a wide variety of sizes, cost and energy densities. Its biggest strength as a power supply was its versatility. It allows us to be able to pick and choose what would be the most beneficial for the Pill^3 at a low cost. For our design, picking a power source with a high energy density was needed. High energy density allows for our device to have a longer life cycle. The use of batteries also allows our device to become portable, making it easier to move around and display it anywhere. The main issue we faced with batteries was the need to replace them when the batteries lost the ability to generate ions into the device or simply ran out of power. Having to keep changing batteries wasn't ideal for the consumer or the environment, as many people around the United States do not dispose of them properly. It could lead to frustration if the consumer has difficulty opening the casing and possibly damage the device if not careful. The Pill^3 was also meant to be stationary. The need for portability wasn't necessary. While batteries would have been a good choice, other power supplies could be more beneficial for our device. However, batteries would still be kept under consideration for our device.

4.2.2 Solar Panels

Solar panels are used in a wide range of products. Being one of the most eco-friendly sources of power, solar panels are also very sturdy. Having solar panels be the main source of power for the Pill^3 comes with a multitude of benefits. Unlike the batteries, solar panels have a lifetime of roughly 30 years, are rugged and are good for the environment. On the days where the solar panel isn't exposed to light, it was able to survive on stored power. However, this type of power supply isn't ideal for the Pill^3. Our device was meant to be used in the comfort of your home, meaning that it would not be exposed to sunlight depending on where the device was placed. While solar panels are able to work under artificial sunlight, it isn't as efficient as sunlight. Due to this, solar panels would not be a sufficient source of power for the Pill^3.

4.2.3 Combinational Power Supplies

An idea that we highly considered using was a mix of two different types of power supplies. Having both solar panels and batteries on the Pill^3 brings forth a plethora of benefits. The solar panels would extend the lifetime of the batteries as it would act as the primary source of power. By being able to store power, it would increase its energy independence and make up for where the other power supplies fall short. However, the same issues arise when considering this idea. Since our device was meant to be kept inside, a lack of sunlight would cause the solar panel to render basically useless. Artificial light wouldn't add much to the solar panel. This would cause the battery to remain the same, causing the consumer to replace the battery when its life runs out. It would also add too many unnecessary components to our device. A more reliable source of power could be used instead.

4.2.4 AC/DC Converter

The final type of power supply we would be exploring was the AC/DC converter. AC/DC converters have a wide range of benefits. It is a reliable source of power, has great energy storage, is rugged and could have a long lifetime. Since we design the overall build of the AC/DC converter, we are able to adjust it as we need. Although it's a good option, AC/DC converters have a few issues. In order for the device to work, we would need an outlet so that the converter could extract its power from. Meaning that the placement of the device was limited within the area in which the cord extends. Since the device was also developed in the United States, outlets across the world would not fit our design since we are following the plug standards set by the United States. Additional adapters would be needed in order for the consumer to use it if in another country. The box in which the AC/DC converter was stored was also an issue. It took up space in the consumer's area, making it annoying to deal with. With it all being said, we would be using an AC/DC converter for our device as it was one of the most reliable sources.

4.3 Motors

When researching potential motors for the Pill^3, several qualities were determined to be most relevant. These qualities include the power usage of the motor, the rated weight in kilograms that the motor could reliably drive, the overall size of the motor and motor housing, and the amount of precision available when controlling the motor. For our design, a motor needs to be low power (consuming below 6 volts), high torque (capable of rotating up to 7 kilograms), fit within the casing of the Pill^3, and be able to be controlled to within 5 degrees of rotation. In order to determine a motor that met all these qualifications, several types of motors and specific motor brands were examined.

4.3.1 DC Motors

Direct Current (DC) motors come in a variety of torque configurations, power usages, and sizes, so finding one that met these usage criteria would be simple. The area of particular strength for DC motors was their ability for continuous rotation and their ease of use. Typical DC motors require only two wire inputs, one for power and one for ground, and would run for as long as power is supplied. This simplicity was reflected in the price of DC motors, which tends to be lower than the other two types of motors examined. The main issue that arises from using DC motors was their lack of reliable precision. It was Difficult to control the exact number of rotations or the size of the steps it takes, and was thus more suited for applications where a high RPM was needed. As such, DC motors would not work for this project.

4.3.2 Servo Motors

Servo motors integrate DC motors in a way that allows for more precise control over the degree of rotation while still operating within the same range of power. By including a gear box, potentiometer, and a control circuit with a DC motor, servo motors could track and control the exact position of the motor as it rotates, and could quickly start or stop the motor to make minute changes. However, the added precision also increases complexity and reduces flexibility. Servo motors have 3 input wires, one for power, one for ground, and one for control commands, which makes it slightly more difficult to set up than a DC motor while allowing for significantly more control. These types of motors also trade the continuous 360 degrees of rotation for a more controlled but smaller 180-270 degrees of fixed rotation. Since our device would be using this motor to rotate a disk within a specified angle, a servo motor would most likely be the type of motor used for the Pill^3.

4.3.3 Stepper Motors

The final type of motor examined was stepper motors, which use a completely different method of driving the gear shaft than DC motors. Rather than using a coil of copper wire to rotate, the stepper motors use electromagnetic gears that could precisely control the position of the motor through subdivisions called steps. However, the increased precision drastically lowers the speed at which the motor was able to operate and increases the complexity twofold, with input and output wires ranging from 4 to 8. The improved positional precision also increased the price per unit of this motor, making it slightly more expensive than similarly rated servo motors. Since the Pill^3 needs precision and torque, speed was irrelevant to our design. As such, this type of motor could possibly be used in our device.

4.4 Microcontrollers

A microcontroller also known as an MCU is an integrated circuit designed to control the actions of other products in an embedded system. This includes, but doesn't limit to sensors, LCDs, motors, etc. There are many different types of microcontrollers. Each microcontroller has similar capabilities in the sense that it could control other devices in an embedded system, but depending on the needs of the user, some microcontrollers would work better than others. Some examples of these would be ESP 8266, MSP microcontroller and an ESP-32 cam. For our device, we would use a microcontroller that has bluetooth and wifi enabled in order to communicate with our mobile application. All extra capabilities of an MCU are based on the chip integrated or piece of hardware put in.

The main component of the Pill^3 was the microcontroller since this device was the one that would put all the other components and devices together. When looking for microcontrollers we had to have in mind the needed features as well as the possible applications of it. These microcontrollers need to have wi-fi capabilities since for our application of the device application a firebase and cloud system would be used to save the user's pill schedule information. Based on this, the microcontroller needs to be able to communicate with the other components as well as the app, for the Pill^3 to work as it was designed. When researching for microcontrollers we find a wide variety of them, in fact we had to analyze each of these kinds of microcontrollers as they could be potentially used for our design.

4.4.1 PIC Microcontroller

This microcontroller is commonly used for computer robotics, development of various electronics and similar devices. PIC stands for Peripheral Interface controller, this controller was based on hardware computing architecture, the programming works based on different registers to increase the amount of inputs and outputs, this device has a built-in data memory, data bus and a dedicated microprocessor for preparing any kind of input or output methods and purposes. When researching for microcontrollers this was one of the devices that were first considered since it meets the minimum criteria for our design. However, we have limitations on the I/O pins, not only that, but the board works with a specific cloud service which could restrict us from using other services since the board was programmed to use these specific programs. Nevertheless, this board created by Microchip technology could be considered in the development of the Pill^3 device.

4.4.2 ARM Microcontroller

The ARM microcontroller is an advanced reduced instruction set computing machine; most of their chips are 32-bit reduced instruction set computers. ARM stands for Advanced RISC Machine, these processors are mainly used in portable devices like cameras, cellphones, home networking modules and many other embedded systems. The advantages of these MCU units was that their power consumption remains quite low for the performance of it. Even though the ARM processor would be convenient to use there weren't as many features for our needs. The ARM microcontrollers focus mainly on portable devices, in our case the device would remain stationary, so that feature would be rather useless for the Pill^3, on top of that there aren't as many

Wi-fi applications built in the board. When researching for an ARM microcontroller there were not as many boards as we wanted, making our search rather difficult. Due to this, we decided on not working with an ARM microcontroller.

4.4.3 CC430 Microcontroller

When looking for microcontrollers we looked specifically for the CC430 from Texas Instruments which in previous classes we have been exposed to some of their products and most of our experience lays on the TI microcontroller applications. A CC430 microcontroller is an ultra-low-power microcontroller system-on-chip with integrated RF transceiver cover consisting of several devices that feature different sets of peripherals that are commonly targeted for a wide range of applications. These kinds of microcontrollers align with the Pill^3 needs since this microcontroller has Wi-Fi connectivity and interface to be used to program the microcontroller was efficient and there was a vast variety of application and user guides that better explains the board functionality. Because of this we considered this microcontroller for the development of the Pill^3 device.

4.4.4 AVR Microcontroller

Lastly for the microcontroller selection we researched on the AVR microcontrollers. An AVR microcontroller is a device with many applications as an embedded system. These are commonly used on education embedded applications; most of these chips have been included in the Arduino line of open hardware development boards. These chips give us a wide variety of applications per board. Like it was mentioned previously our main interest when selecting this microcontroller was that it has Wi-Fi capabilities since that would be the main form of communication with the app that would be developed for the Pill^3 device. A lot of companies use these kinds of microcontrollers for the development of most of their products since it gives a lot of features that could be implemented in different applications. For our sole purpose an AVR microcontroller would be a great selection because aside from the hardware features and the extensive amount of digital pins available, the programming involved for this board was very simple to develop and use. This microcontroller uses a high level language for their programming portion, although it could be challenging, it would have a better

4.5 Speaker

A speaker is a device that takes in analog audio signals and converts it into air vibration in order to make sound waves. Also a type of transducer, the speaker takes in audio input from a source, either in AC or DC, and amplifies electromagnetic waves into sound waves. In the case of a digital speaker, the digital input must be converted into an analog signal and follow the similar trend in order to create the sound waves. Most speakers include the following: A cone, coil, magnet and chassis. Speakers also come in a wide range of shapes and sizes. For our device, we would be using a miniature speaker since our device would not need anything grand and would fit the snug size of our product. Cost, sound and efficiency are the main components that we would be looking at, The purpose of the speaker was to act as a secondary notification, as it would notify the consumer when the pills have been dispensed.

4.5.1 Piezoelectric Speakers

Piezoelectric speakers, also known as buzzer and crystal loudspeakers, are made from piezoelectric material. It utilizes the piezoelectric effect to create sound. When a voltage is applied onto the material, it creates a mechanical motion, moving the diaphragms and resonators to create sound. Due to the material used, a big advantage of using a piezoelectric speaker was its resistance to overloads. Meaning that they are more durable than most speakers. Its low cost and snug size makes it a good candidate for our device. The issue we faced with a piezoelectric speaker was its sound quality. It was not as good as other types of speakers, but still provides good volume. We would be considering this type of speaker for our device.

4.5.2 Plasma Arc Speakers

Plasma arc speakers are one of the least type of speakers used on the market. It takes in high energy electrical plasma to vary air pressure in an amplifier to create sound. It creates a unique sound and adds an amazing visual. Since the plasma arc speaker could be built or bought, the cost also varies. The downfalls of a plasma arc speaker was its practicality, size and sound. In order to implement this speaker onto our device, we would have to extend the size of the design and add too many unneeded components just to produce sound that isn't very high in quality. Other traditional speakers could be used for our device that do not have these issues.

4.5.3 LoudSpeakers

Loud speakers are one of the most used types of speaker on the market. Since this speaker was so versatile, there was a wide range of speakers to choose from. The cost heavily varies depending on the size, brand and quality needed from the user. Loud speakers also allow customization. Bluetooth could be enabled or could be accessed via a wire. Its versatility and clear sound are its biggest advantages. The biggest drawback would be the testing aspect. We would need to look for a loudspeaker that fits our device and has a good sound quality for a low cost. Loud speakers would be highly considered for the Pill^3.

4.6 Displays

One of the many features our device brings was the addition of a display. The need for a display was very useful as it was what would be our secondary source when notifying the consumer when the pill was dispensed, show the count of pills being dispensed and display a countdown of when the next batch was to be dispensed. We take into account many different factors such as cost, size and usability when deciding on which display best fitted for our device. We would need a display that was clear for the user to see and give visual signals of our notifications. We explored several different types of displays in order to determine which was best for the Pill^3.

4.6.1 LCD

An LCD or a liquid crystal display is a device that could be controlled in an embedded system. It Is filled with liquid crystals in order to propagate images on the screen. LCDs are used all over the world for a wide range of applications, some of these include, but aren't limited to televisions, calculators and smartphones. Some benefits of utilizing an LCD display was the simplicity and versatility of it. Many LCD displays function perfectly with many different types of microcontrollers and come at a low cost. In addition to these, some LCD displays also produce very little heat and are power efficient meaning that little energy was required in order to have it operate properly. The issue we face stems from the limitations that the LCD holds . Certain LCD displays have uneven backlighting, meaning that some parts of the display could be brighter than others. It also has limited view angles, making it hard to get the full picture based on where you are standing. With this in mind, we still consider the LCD display to be one of our top candidates for the Pill^3.

4.6.2 LED

An LED display or Light emitting diode is a panel that utilizes an array of light emitting diodes, also known as pixels, to display an image. Like the LCD, LED displays are used in all sorts of electronics and work well with many microcontrollers. LED displays are superior to LCDs when it comes to its energy consumption, life span and visual prowess. The issue we faced with LED displays was its price and size. Since it was a very high quality piece of hardware, LEDs could have a high cost and while having a higher visual quality was always better, our device would not be in need of it since we would only be displaying a simple message. Many LED displays are also limited in size. Finding a small one to fit our device would limit us. For these reasons, the LED display was not ideal for the Pill^3.

4.6.3 OLED

OLED or Organic light emitting diodes utilize an organic compound to emit light as a response to electric current. They are used in many electronic devices and could pair well with several microcontrollers. OLEDs have several advantages over LCDs and LEDs. Due to its organic layer, OLEDs are thinner, lighter and more flexible than LCDs and LEDs. OLEDs are also brighter than LEDs and do not require backlighting like LCDs. Due to all of these factors, OLEDs require much less power than its competition. The OLED seems like a perfect candidate for our device, but the fragile nature of the OLED, short lifespan and cost makes it non-ideal for the Pill^3. We would keep the OLED under consideration, but other devices could fit our device much better.

4.6.4 Seven Segment Displays

Seven segment displays are one of the most common types of displays. The biggest advantage in using the seven segment display was its simplicity. Since the display only consists of four input leads, there was little risk of having malfunctions. It could withstand a wide range of temperatures, has a low cost and was very effective. The biggest issue we faced with the seven segment displays was also its simplicity. While the Pill^3 would not require high quality graphics, it would need to output more than just a number. For this reason, the seven segment display would not be considered for our device.

5.0 Strategic Components and Part Selections

For this section we would go further into details as to which parts were selected for our device. We would compare and contrast each specific device. Many factors were taken into account when comparing which piece of hardware was selected for use.

5.1 Sensors

For the Pill^3 numerous sensors would be utilized. At first the ultrasonic sensor was one of the main components to be utilized, however, it did not meet the minimum criteria for our design. The idea of this sensor was to detect the pill as it was falling into the dispensing area, however, the reaction time for the ultrasonic sensor was very slow. Due to this we decided on looking for Infrared sensors. There are two kinds of infrared sensors that could be added into the design, based on various factors and usage, which would be discussed on the next pages. Other sensors that were analyzed were the weight and temperature sensor which would be incorporated in the final design.

5.1.1 Obstacle avoidance IR Infrared Sensor

This sensor has been used in the past as a robot obstacle avoidance, line count, obstacle avoidance car, etc. For our purposes this sensor would be the one to determine when a pill has left the storage area. This sensor has one light that would be always on, if no object passes through the light no reflection would be detected by the IR sensor receiver, if an object, in this case a pill, passes through the IR light transmitter then this would cause a reflection and it would send a signal to the MCU unit which keeps the track of the pills of each of the storages. This sensor has an effective distance range of 2 ~ 30cm, and it was able to properly function with 3.3V – 5V. Its detection angle was 35 degrees default, and it could be adjusted using the potentiometer. This sensor could detect small objects and it could be directly connected to the MCU unit if needed. We have the option to screw the sensor into the board since it has 3mm holes on each corner, and its response time was less than 1ms. In terms of price, they are affordable and could be bought in bulk if needed. Below we would see a figure that represents the application of the Pill^3 dispensing system.

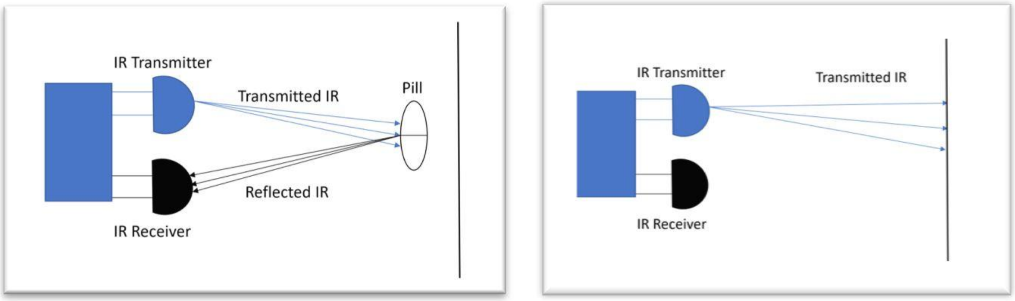


Figure 2 : IR Sensor

5.1.2 IR Break Beam Sensor

An IR Break Beam sensor works very similar to the Obstacle avoidance IR sensor; the difference was that the sensor was smaller. This sensor could be found in a lot of daily products, mainly on the bathrooms where it would detect whether to turn on the foset and when to turn it off. That was one of the most common applications, however, this sensor tends to have a large margin of error since it won't function unless it detects the object. For the Pill^3 the pill could be falling at a fast rate and there was no guarantee that the sensor would detect it. The sensing distance was approximately 50cm, it could be powered using 3.3V – 5V, the transmitter angle cannot be changed internally in the sensor but it's 10 degrees, the response time was greater than 2 ms and the cable was 250mm long. After careful analysis we decided not to work with this sensor due to the slow response time and the restriction on the wire length.

Table 2: IR Sensor Comparison

Sensor	**IR Break Beam Sensor**	**Obstacle avoidance IR Infrared Sensor**
Operating Voltage	3.3V - 5V	3.3V - 5V
Sensing Distance	50 cm – 45cm	2cm – 30cm
Transmitter angle	10	35 (adjustable)
Response time	<2ms	<1ms
Price	$ 5.95	$ 8.29

Analyzing further our selection we could see that the best option for the Pill^3 dispenser would be the obstacle avoidance IR sensor. This sensor was able to provide us a faster response rate as well as numerous adjusting options that could be applied when programming the MCU unit. Although the sensor could be on the expensive side the price was based on approximately ten sensors, so realistically speaking one sensor was actually $0.83 dollars which was an affordable price, while the IR break beam sensor was about $5.95 dollars, due to this, the obstacle avoidance sensor was a better option for the development of the Pill^3 device.

5.1.3 Temperature Sensors TMP36

When we were at the very first stages of the design we came to the conclusion that a temperature sensor would be needed. Based on the research made on the motors we notice that some of them might generate some heat which would be a dangerous situations since the idea was that the components are all in one area in order to save space, because of this we decided to add a temperature sensor and a fan as a precaution for the customer and the device alone. This sensor would be added into the PCB design, there could be a change as to where specifically to place this sensor but it would be close to the MCU unit.

For the Pill^3 a temperature sensor was an added feature we decided to add. The idea of this sensor was to be able to identify when the temperature was increasing for the board due to the motor usage and activate the fan placed close to the MCU unit. Only one sensor would be needed. The TMP36 could detect temperatures as low as -40C up to 150 C. It was able to operate with 2.7V – 5.5V. This sensor takes around 15 – 20 second to react to the change in temperature which could be a problem since after that set amount of time the components could have been damaged. Because of this we would have to combine all components and observe how much heat was generated from it. These sensors are not expensive, which makes the development of the Pill^3 easier to analyze.

5.1.4 Temperature Sensor LMT85LP

The LMT85LP was similar to the TMP36 sensor in terms of size and applications, the difference was that this sensor was able to detect a change in temperature after 0.4C which could take less than 5 seconds depending on how drastic the change in temperature is. The temperature that the sensor could handle was -50C up to 150C. The advantage of this sensor was that it could operate with as little as 1.8V – 5.5V, for the Pill^3 this could be a good option since it was an affordable component and it has a fast reaction time.

Table 3: Temperature Sensor Comparison

Sensor	**TMP36**	**LMT85LP**
Operating Voltage	2.7V - 5.5V	1.8V - 5.5V
Temperature detection	-40C - 150C	-50C - 150C
Response time	15s - 20s	After 0.4C

Price	$1.40	$0.9

Even though the temperature sensor was an added feature of the Pill^3 device we had to take into consideration possible options for these devices. As we can see in the table above the best option would be the LMT85LP, not only in terms of cost but also based on the rate in which the sensor was able to detect a change in temperature. If the board were to reach abnormal temperatures, the sensor would be able to activate the fan after an increase of temperature of 0.4C. This detection could be almost immediate, however, even if it takes long the sensor would be able to alert the MCU unit that there has been an abrupt change in temperature. In conclusion, we decided to go for the LTM 85LP because it was the most affordable and efficient compared to the TMP36.

5.1.5 FlexiForce Pressure Sensor

Lastly we have the pressure sensors, these sensors would be one of the key components of the Pill^3 since the sensor would be directly connected to the MCU unit so let the system know as well as the app when the user has taken his medication as well as when the device was ready to dispense any schedule pill. The way the sensor will be programmed is as shown in figure 3, the user will be required to place a cup in order to obtain the medicine from the device. The idea was to use only one sensor since the only area that required it was the pill dispensed area.

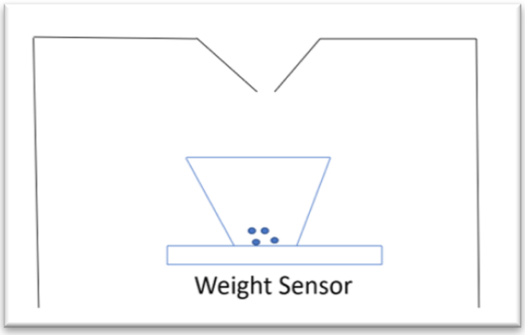


Figure 3 : Pressure Sensor

For Pill^3 we would be utilizing the force sensing resistor that acts like a pressure sensor. Its snug size and weight was ideal for our device. It was a thin and flexible sensor made with piezoresistive material. It was able to measure forces up to above 22 lb, but for our device we would only be detecting the cup when placed on top of the sensor, so the mass would be very minimal. At the time of this report, this sensor was ideal due to its accuracy. As we start testing, we would determine if this was correct for us. However, this would fit our design aesthetically, while also providing the data we need. Its low-cost was also a plus as it makes our device more affordable.

To go further in depth with the pressure sensor, we would take a look at the inner workings. Also known as a weight transducer, this sensor converts a mechanical load, the input, and an electrical signal is given out, the output. This was possible due to the piezoresistors embedded into the device. Piezoresistive materials were what drove this. When the material experiences stress, an electrical signal is given out, thus causing a reading for the signal. Although this sensor was expensive, the features and accuracy are needed for the Pill^3 device.

5.1.6 Force Sensing resistor - Pressure Sensor

Looking at other pressure sensors on the market, the force sensing resistor has similar usages as the flexiforce. It follows a similar aesthetic but it covers a larger area, this sensor provides the desire detection accuracy. For our device, we are in search of a device that could detect 0.101g since that was the accuracy standard weight for most pills, however, we could use the sensor for different approaches for the Pill^3. The big advantage of this sensor was its price since if compared with other sensors this one in particular was more inexpensive, and in terms of applications we are not as limited to the weight the sensor could detect. One of our main goals was to give the user the opportunity to place any cup under the dispensing area. With this device we gave the user the flexibility to use any cup they desire.

5.1.7 FSR04BE pressure sensor

When comparing it to other pressure sensors on the market, the FSR04BE has similar usages as the flexiforce. It follows a similar size and aesthetic, but the FSR04BE doesn't have the accuracy we desire. Its sensing range was only from 20g to 5Kg. For our device, we are in search of a device that could detect 0.101g since that was the accuracy standard weight for most pills. The only big advantage of this sensor was its price since if compared with other sensors this one in particular was more inexpensive, but in terms of applications we are limited to the weight the sensor could detect. One of our main goals was to give the user the opportunity to place any small cup under the dispensing tube/hole, so if they lose the cup it could be easily replaced. We cannot depend on a specific cup weight, not the pills insce some pills are smaller than others and each of them have different weights.

Table 4: Pressure Sensor Comparison

Sensor	FSR04BE	FlexiForce	Force Sensing Resistor
Thickness	0.325mm	0.203 mm	0.508mm
Length	15.80mm	191mm	38.1mm(1.5 in)
Width	7.62mm	14mm (0.55in)	38.1mm(1.5 in)
Sensing Area	5.60mm	9.53mm (0.375in)	38.1mm(1.5 in)
Connector	2-pin male	3-pin male square pin	3-pin male square pin
Weight Capacity	20g - 5Kg	0g - 500g	28g - 10Kg
Price	$7.03	$19.95	$11.98

Lastly, based on the research made for this weight sensor, we have decided to go for the force sensing resistor (pressure sensor) since it was the only sensor that has the most accurate reading. For the Pill^3 it was extremely important that the sensor communicated with the MCU when it detected the pills falling and if the cup had been removed from the dispensing area. We encounter sensors that would work for heavier objects with a weight higher than 28g. For the Pill^3 that would not be a good option since all we needed was the device to be able to detect a cup. An option that came up while researching for this was to have a specific cup for the customers to use. If the user were to lose that cup then it was forcing the user to either get a similar one or purchase a replacement, with the sensor selected the weight of the cup would not affect the user's experience. Our goal was to provide the user the most easy to use and flexible device. Even if the sensor was slightly higher in price it would guarantee us a more accurate reading.

5.2 Power Supply

This section would cover how we would power our device and compare it to the different types of power sources available. We would discuss each candidate based upon functionality, efficiency and why we chose our current source of power. It would go into specific details.

5.2.1 AC/DC Converter

Since the Pill^3 was meant to be a stationary device, a power source such as a battery wouldn't be very effective since it would eventually run out of power. We will, however, still discuss the up and down sides of the several different types of batteries later in the text. We decided to utilize an AC/DC converter since it could be plugged onto a wall outlet, keeping a steady flow of power for a longer lifetime. In order to proceed with our decision, we must follow its standards. In the United States, the standard for voltage was 120 V and 60 Hz for frequency. Power sockets and outlets in the United States are also A or B type. In order to power the Pill^3, we would need an AC/DC converter that follows this since its power source would come from a wall outlet and have a plug that was either A or B.

In the Pill^3, all the components require a source of power. This would include the MCU, sensors, motors, LED screen and other components that would be connected to the PCB. In order to provide this power we must use an AC/DC converter. Originally we were going to build our own converter, but as we progress in our project, we concluded that purchasing one and connecting it to our main PCB would be the most optimal choice. However, we will still discuss how we would've built a converter.

The first step in doing so would be to pick a transformer so that the AC voltage obtained from the wall outlets could be converted. The secondary part of the transformer would then be connected to a full wave bridge rectifier so that the sinusoidal AC waves continuously get positive peaks. Although still positive, there was time in-between peaks that needed to be removed. In order to filter these out, a smoothing capacitor would be implemented. It would store the incoming energy and reduce the voltage drop. This would result in the voltage having a DC output. Finally, output voltage variation was smoothed out with a voltage regulator to have a steady DC output. Below we can see the AC/DC converter we used to connect our PCB to and provide power to our device.

Figure 4: AC/DC Converter

5.2.2 Voltage regulators

In this section, we will dive further into voltage regulators and discuss its importance for our device as well as for the AC/DC converter. Voltage regulators are used in electrical systems to regulate voltage as the name states. It would set a range in which the voltage would have to stay in. This was done since the voltage regulator delivers a constant source of DC power to keep the system constant regardless of input voltage. The purpose of a voltage regulator in our device was to keep the voltage controlled and avoid damaging any piece of hardware. Linear voltage regulators and switching voltage regulators are the two most common types of regulators. We would be looking at these two to see which one was most compatible for our device.

A linear voltage regulator was composed of a linear system. It uses a closed feedback loop to maintain a constant stream of voltage across the output terminals. Some advantages of using a linear voltage regulator was the cost and simplicity of the regulator. They are also very adaptable, meaning that any change in input voltage would produce an output voltage, an output voltage that was almost ripple free on the input. The issue we faced with voltage regulators was its efficiency and heat generation. Due to the voltage drop across the system, this could cause the linear voltage regulator to heat up. We would have to be cautious of this issue if we were to use this system for our device. While there are ways to manage temperature, it would be in our best interest to try and search for a different system.

Like the name suggests, a switching voltage regulator was a voltage regulator that uses a power switching element to convert the power supply into a voltage. This voltage was then controlled by a capacitor, inductor and diode. A FET overlooks the output of a feedback loop to ensure that it is being regulated. By doing so, it was possible to supply voltage more efficiently with less heat. This was one of the main advantages switching voltage regulators have over linear voltage regulators. The issue we faced with the switching voltage regulator was its high levels of noise and complex design. The cost of this regulator was also high due to this.

When comparing both systems, each has benefits to their own unique design. While the linear voltage regulator was much simpler and cheaper than the switching voltage regulator, the efficiency offered by the switching regulator was much greater. Utilizing a linear regulator could cause more harm. For this reason, we decided to use a switching voltage regulator for our device.

Table 5: Voltage Regulators

----	**Linear Voltage regulator**	**Switching Voltage regulator**
Inverting Buck Boost	Possible Not Possible Not Possible	Possible Possible Possible
Efficiency	Low	High
Linear Regulation	0.02% - 0.05%	0.05% - 0.1%
Load Regulation	0.2% - 0.1%	0.1% - 1.0%
Output Ripple	0.5 - 2mV$_{rms}$	25 - 100mV$_{pp}$
Input Voltage Range	±10%	±20%
Output power	Varies on thermal design	Large Power
Noise	Low	High
Design	Simple	Complex
Parts needed	Not many	A lot
Cost	Low	High
Other Pros	Great for low power devices	Small size Flexible applications
Other Cons	Larger, heavier size	----

5.2.3 Batteries

Other sources of power that we considered were the different types of batteries. While it was efficient for other products, it wasn't the best choice for our device. While solar panels in combination with batteries are good for the environment, it was not ideal. The Pill^3 being an automated dispenser would most likely be placed inside a home. This would reduce the power obtained from the sun, not providing our device much power to function. The three most commonly used types of batteries are Nickel Cadmium, Nickel Metal Hydride, and Lithium ion. We would be discussing how each operates.

Nickel Cadmium batteries are one of the most common batteries out in the market. It was used in a wide variety of products and holds a lot of merit. Its inexpensiveness, fast recharging, long shelf life, and rugged feel was what makes this so widely used. However, this battery also has its downsides as it has low charge capacity, low energy density and needs to be disposed of properly as cadmium is toxic. We considered using Nickel Cadmium for our device, but needed a more reliable source of power, thus we moved on to the next source.

Nickel Metal Hydride batteries are another common type of battery. Its high density power, standard size, moderate life cycle and rechargeability makes it popular amongst the masses. This was also less toxic than Nickel Cadmium, but it discharges at a faster rate and generates more heat from its peers. Although still fairly cheap, we considered a different type of battery source for our device.

Lithium ion batteries are the most popular of the two mentioned above. Its high energy density, self-discharge, low maintenance and wide range of variety was what makes this battery so popular among electronic devices. Its cost, low durability and lack of resources was what hindered the batteries' overall use. It could have been a good source of power for our device, but the cost of testing it multiple times wouldn't have been beneficial for us. Thus we decided to move on to a different power source.

As shown in table 6 below, each battery has its own merit and was used for their specific needs. While it would have been possible to apply these sources for our device, an AC/DC converter eliminates the downfall of these other power supplies. Although portability isn't possible, our device was meant to be stationary. An AC/DC converter could provide enough power for most devices, it's just based on how it was built. Its durability allows for a longer lifetime and its affordability allows us to control how much was put into the product. For the time being, the AC/DC converter was our main source of power, but this was prone to change as we started to build and test our device to see how it could be improved.

Table 6: Battery Comparison

Battery Type	**Capacity AA Cell**	**Voltage**	**Self- discharge Capacity after 1 year storage**
NiMH	2,700mAh, rechargeable	1.2 V	50 % -85 %
Regular alkaline	2,800mAh non-rechargeable	1.5 V	95% 10 year shelf life
Reusable alkaline	2,000mAh; lower on subsequent recharge	1.4V	95 %

Lithium	2,500-3,400mAh (non-rechargeable)	1.5V	Very low 10 year shelf life

5.3 Motors

Since either a stepper motor or a servo motor could be used to achieve the design specifications, two different models of each type were researched and compared. The companies through which these parts would be obtained were thoroughly vetted to ensure that they comply with all constraints relating to pricing and ethics. MOONS' stepper motors were chosen for their well documented quality and in-depth datasheet analysis, which allowed for clear understanding of the components examined. ReadytoSky and SunFounder were chosen for the servo motors, as both companies have clear records and high reviews, and the motors they offered were scaled well to fit with the design of the Pill^3. The information included in the research report below was from the available datasheets and promotional material for each type of motor. While not every useful specification was listed, there was enough information available for each motor to make an informed decision on its usefulness or relevance to the project. The information below was condensed into a tabular format and discussed among the team members before a decision was reached.

5.3.1 MOONS' ML23HS4P4150 Stepper Motor

This motor was a quick response, high efficiency, torque geared stepper motor with step sizes of 1.8 degrees. It has four Input/Output (I/O) terminals, was rated for 1.5 A of current and has a holding torque of 1.2 Newton-metres. It easily meets the requirements set forth for precision and size, as it was about 277% more precise than required. However, it requires 24 V of DC voltage to operate at its lowest performance spec, meaning the power supply and AC/DC converter of the Pill^3 would need to be improved to accommodate this increased draw. Additionally, each unit of this stepper motor was priced at $45.60, which accounts for 15.2% of the total allotted research and development budget and 22.8% of the total manufacturing budget for a single unit.

5.3.2 MOONS' MS16HS4P4100 Stepper Motor

This stepper motor was similar to the other MOONS' motor examined, however its smaller size and lower power draw make it more relevant for the Pill^3. It has a 1.8 degree step angle, 4 I/O terminals, was rated for 1 A of current and has a holding torque of 0.27 Newton-metres. Although it has a lower holding torque than the larger motor above, it retains the same degree of accuracy while requiring less power to operate. This motor needs 12 V of DC voltage to operate, half of the larger option, and was priced at $36.70. It still requires more power than our team had planned for a motor and was slightly over the expected budget for this component, but it meets all the required specifications above.

5.3.3 ReadytoSky TD-8325MG Digital Servo Motor

This low power, high torque servo motor has an operating rotational range from 0 to 180 degrees, an operating speed of 60 degrees in 0.15 seconds at 5 V, a stall torque of 25 kg/cm, and has 3 I/O terminals. To control the speed and direction of the motor, pulse width modulation (PWM) was used within a range of 500 to 2500 microseconds. Compared to the stepper motors, it has significantly lower power while still retaining a fairly strong holding torque. While the exact resolution of the motor was not provided by the available datasheets, it was assumed to be at least below 5 degrees, since it was controlled through PWM and thus could be controlled accurately. Additionally, the price was significantly lower than that of the stepper motors with each unit costing only $18.99, or 9.5% of the total manufacturing budget.

5.3.4 SunFounder SF3218MG Digital Servo Motor

This servo motor is low power, high torque, and has an operating rotational range from 0 to 270 degrees. This motor requires 5 V to operate and has 3 I/O terminals, similar to the ReadytoSky motor, but has a lower stall torque value, 20.5 kg/cm, and a slower operating speed, 60 degrees in 0.18 seconds. However, the lower speed of the SunFounder motor means that it was more accurate, if only slightly. It was the cheapest motor researched, with a price per unit of $13.99 or 7% of the total manufacturing budget, but it meets all the specified requirements.

5.3.5 Adafruit NEMA-17 Stepper Motor

The Adafruit stepper motor solves the biggest drawback of the MOONS' stepper motors in regards to this project, which was the price. It nearly ties the SunFounder servo motor for the cheapest motor at $14.00. Similarly to the MOONS' MS16HS4P4100 stepper motor, the Adafruit motor was rated for 12 V, has 1.8 degrees of accuracy, and 4 I/O terminals. Aside from the price, it has a slightly lower rated torque at 0.2 Nm, and a slower RPM at 50. However, the torque for this motor was still within acceptable range, and the slow RPM was not an issue for the proposed usage of the stepper motor in the final design of the project.

Table 7: Motor Comparison

Motor	**MOONS' ML23HS4P4150**	**MOONS' MS16HS4P4100**	**ReadytoSky TD-8325MG**	**SunFounder SF3218MG**	**Adafruit NEMA-17**
Rated Voltage (V)	24	12	4.8-6.8	4.8-7.2	12
Rated Current (A)	1.5	1	--	--	.35
Rated Torque	1.2 Nm	0.27 Nm	25 kg/cm	20.5 kg/cm	0.2 Nm

Output Shaft Diameter (mm)	6.35	5	--	--	5
Degrees of Rotation	360	360	180	270	360
Degrees of Accuracy	1.8	1.8	5>	5>	1.8
Weight of Motor(kg)	0.48	0.21	--	.56	--
Frame Size (mm)	56.4	39	--	20.5x40.5	42
Frame Length (mm)	45	33	--	40	31
I/O Terminals	4	4	3	3	4
Price ($)	45.60	36.70	18.99	13.99	14.00

The above table organizes the important qualities of the researched motors in a way that makes it easy to analyze. Between the examined servo and stepper motors, our team decided to use a stepper motor in the project. Since the pill dispensing mechanism was intended to rotate a full circle, the limited degrees of rotation of a servo motor were actually disadvantageous. Using the table above, the Adafruit stepper motor was chosen over the MOONS' stepper motors to be used in the medicine storage cylinder of the Pill^3. The Adafruit stepper motor was the right balance of price and precision, while still accounting for the torque needed to move the cylinder.

5.4 Speakers

For our device, speakers would be used as one of our many ways to notify the user when the pills have been dispensed. We consider many candidates for our device. While piezoelectric speakers would work for the Pill^3 , loudspeakers are what we would be discussing in the sections below as it gives us a broader range. Cost, functionality and efficiency was what we would be looking at when deciding on the speaker.

5.4.1 CQRobot passive speaker

This was a small passive speaker with JST PH2.0 interface. It has a frequency range of up to 10,000 Hz, could operate in a wide range of temperature, weights 24g and has a small build to it. It requires a power rate of 3.0 W/ 3.5 W. Since it comes in a number of flavors, we could order the one that meets our needs. It was also suitable for most devices, including our arduino. Set up was easy and only required an audio decoder module. Costing only 3.99, this was a great speaker for what it provides. The only drawback was the size of the speaker, it was too tall but was still a great candidate for the Pill^3.

5.4.2 MakerHawk

The MarkerHawk 2pcs arduino speaker was a smaller speaker than the CQRobot passive speaker. It has a frequency response of 500 – 20kHz, weights 15g and has a very small build to it. It requires a power rate of 3W. Just like the CQRobot passive speaker, it requires an audio decoder module. The drawbacks for this speaker was its price and volume. Costing $22.01, it was too high for what was provided. Cheaper alternatives for better quality could be found. The MakerHawk also provides a low volume but good quality sound. This was not ideal for the Pill^3 as we would need a device that even the hard of hearing could use.

5.4.3 Uxcell

The Uxcell was a mini circular loudspeaker. We consider using this speaker for our device based on the cost, size and efficiency. When comparing the audio to its competition, the Uxcell does fairly well for its size. Due to its small stature, not much power was needed for this device. The drawbacks we faced with Uxcell was the functionality. Although we were able to acquire four for $9.90, according to sources it was prone to break and be loose on the breadboard. Any malfunctions are not ideal for the Pill^3, thus we would not be using the Uxcell speaker for our device.

5.4.4 Gikfun

Gikfun 2 stereo woofer loudspeaker was one of our highly considered speakers for our device. It was the second smallest out of all the speakers we found. Its snug size was ideal for the Pill^3. In comparison to its competition, the Gikfun 2 loudspeaker also provides a clear sound for its price. It consumes the same amount of power and weighs relatively the same as its competition. Functions well with our arduino and we are given two for the price of $9.99. We would be using the Gikfun for the Pill^3 as it meets our standards. This was prone to change as testing started to occur for our device. We would only use one of the two speakers as only using one provided enough sound.

Table 8: Speaker Comparison

Speaker	**CQRobot Passive Speaker**	**MakerHawk**	**Uxcell**	**Gikfun**
Size	70mm x 31mm x 16mm	31mm x 28mm x 15mm	28mm x 6mm	30mm x 2mm
Cost	$3.99	$22.01	$9.90	$9.99

Weigh	24g	15g	27g	25g
Power Rate	3.0 W / 3.5 W	3.0 W	1 W	3 W
Impedance	8ohm plus/minus 15% at 1kHz	8 ohms	8 ohms	4 ohms

When looking at the data provided by the table and research done, we came to the conclusion to choose GikFun 2 stereo woofer loudspeaker. Size was not much of a factor when picking our speaker since it was all relatively the same. This could also be said about the weight and impedance of the device. It all came down to sound quality and cost. Although it was not the cheapest option, it was the best option when doing an audio test. For this reason, we are picking this option for the time being. It might be prone to change as we start to test our equipment and devices more.

5.5 Displays

The display would act as one of our secondary forms of notifications. For our device we want to ensure that the consumer has multiple subtle reminders so that he/she would not forget when to take their medication/vitamins. Due to its sole nature of being subtle, we do not need a high quality graphic being displayed nor a simplistic one that only shows the bare minimum. For this reason, we decided to use an LCD or LED display for our device. LCD and LED displays come in a wide variety of shapes and sizes. This was ideal as we test out what would be the most beneficial for our device and look for the most appealing. We would be looking at four different types of LCD screens. Each has its own merit and a detailed comparison would be made in order to help us decide what was best for our device. The screen was prone to change as we test out our equipment.

5.5.1 MakerFocus

The MakerFocus 2Pcs I2C LED display was one of the displays we're considering using for the Pill^{3}. Its bright display, low cost and size makes it ideal for the Pill^{3}. Costing only $12.99 for a pack of two makes this a highly sought piece of hardware for our device. It was simple enough to display what we want to show, but at the same time not as simple to have limited functionalities. It also pairs well with our microcontroller. Having an operation temperature of 40 ~ 85 ºC makes it very versatile based on the consumer's living conditions.

5.5.2 SunFounder

SunFounder LCD1602 was another one of our candidates for our device. When comparing it to the MakerFocus, we have a bigger screen and a higher resolution at a lower price. It has a working voltage of 5 V but could be adjusted with a 50k Ohms potentiometer. Since this was an LCD and not an LED, the picture quality was not the greatest and this was where the devices differ. However, it does have the same functionalities as the MakerFocus 2Pcs I2C LED display.

It also connects well with our chosen microcontroller. We would be considering using this display for our device, but it may not be in our final draft.

5.5.3 WaveShare

The 2.8 inch Touch LCD shield by WaveShare is a resistive touch, arduino compatible and customizable display. This display allows the user to insert a micro SD card, is backlight adjustable and has lower power consumption than its competition. With its high resolution and vibrant colors, the LCD display was a top candidate for our device. The issue we faced with WaveShare was the price and quantity. Unlike the MakerFocus and SunFounder, WaveShare only provides us with one screen for 2 - 3 times the cost of the other candidates. Although it has very good quality, it was not necessary for the Pill^3 as our display would only show basic reminders to the user.

5.5.4 HiLetGo

The HiLetGo 2Pcs HD44780 1602 LCD display module was a widely used display. Although it appears as your standard LCD display, the LCD1602 has an I2C port integrated to solve common issues that many displays bring forth. It is simple to use, has two bidirectional data lines, a serial data line and a serial clock. Although it does not have high quality graphics like the other displays that we have seen, it was a good fit for our device since it provides all the requirements that we are looking for.

We had a wide range of choices to pick from. Each display would be functional for our device and deliver amazing results. When comparing the number of pins, power rate and sizes, the majority was the same. After looking at the options in table 9, we choose the MakerFocus 2Pcs I2C LED display due to the quality and simplicity of the device. Its price was the median of all the options. Setup was simple and was compatible with our microcontroller. This was our choice of display for the time being, but it was prone to change as we started to build and test our device.

Table 9: Display Comparison

Display	MakerFocus	SunFounder	WaveShare	HiLetGo
Size	38mm x 12mm	80 mm x 36 mm x 12mm	2.8 inch	88mm x 35mm x 11mm
Cost	$9.59	$5.99	$23.99	$8.49
Power rate	3.3 ~ 5 V	3.3 ~ 5V	3.3 ~ 5V	5 V

Resolution	128 x 32	128 x 64	320 x 240	-
Number of Pins	4 Pins	9 Pins	9 Pins	9 Pins

5.6 Processor and Memory

This section would go over various microcontrollers that could be utilized for our device. There are specific functionalities that we are considering for the device, one of them being wi-fi capabilities in order to connect with the app we implement. Our main goal was for the user to connect with the device through their mobile device which would be the main route of notifications for the user. We also considered if there were to lose their mobile device, a secondary notification system would be the speakers and the LCD in front of the device. All of this would be embedded into the MCU unit we decide on utilizing

For our device we considered four main processors, the main goal for the processor was to take an storage information from the user as well as have a timer implemented depending on the input given, not only that, but it also has to be able to detect the weight of the pill once its dispense to determine if the user took the pill or if it was skipped. We would have multiple sensors connected to it which would make the device more efficient. The processors being considered are: Arduino, TI microcontroller, PIC and ATMEL.

The main things to be consider when selecting the microcontroller to use was that the processor have enough I/O lines, was able to communicate with both sensors and the app, and make sure the device alone has enough RAM and ROM. These mentioned before would be evaluated based on their design and evaluating how it would be applied.

5.6.1 Arduino Uno Wi-Fi REV2

The first MCU unit to be considered was the Arduino. The Arduino was an open-source electronics platform based on a simple software interface and user-friendly hardware. The Arduino board was capable of reading inputs like turning on/off an LED, storing information, activating a motor, sending digital signals, etc. The Arduino has its own interface based on processing. The benefit of the Arduino was that it was not an expensive device, these MCU could go for less than $50, additionally it has the Cross-Platform capabilities, and it has a simple and clear programming environment. Although the Arduino tends to be considered a MCU mostly for learning processes it has been shown that a lot of other industries are using these devices since it's much simpler to use mostly when it comes to simple applications and on top of that the IDE gives the developer the ability to program and test the code without running into any problems in the process. This MCU unit was considered to be part of the AVR processors.

For our purposes the Arduino would be ideal since this microcontroller unit has wi-fi capabilities which was one of the most important things in our implementation of the app. Our application would be connecting with the device through a wi-fi connection hence the importance of this specific feature. Based on the research done, the ARDUINO UNO Wi-Fi REV2 would be the ideal MCU. The board has a total of 14 digital input/output pins, 6 analog inputs, one USB connection, a power jack, an ICSP header, and a reset button. This board could be powered using an AC adapter or something as simple as a battery. This board has a microchip ATmega4809 microcontroller, even though it contains this chip, the board has a compatibility layer included in the core which allows the Arduino to run without a problem. Evaluating all the features of the Arduino could be part of the development of our device.

The Arduino UNO Wi-Fi REV2 has a CryptoAuthentication device embedded in the board; this chip provides the developer a safer key storage that could be used to implement a wide variety of encryption protocols. This chip was commonly seen in a lot of the MCU units since it was one of the most reliable encryption systems. For the Pill^3 this component would be needed as we are working with the user-sensitive information, if the information were to be exposed to unauthorized users that could be a potential life threat as some of these medications need to be delivered to the user in a specific time or day, as it was programmed in our system. Hence having this feature would be beneficial to the customers of the Pill^3 device.

The Arduino environment has its own IDE. This Integrated Development Environment was a cross-platform application for any of the most common operating systems. The advantage of this IDE was that it uses functions from the C and C++ libraries, for which most of the developers are familiar with. The Arduino IDE offers an extensive library of functions that could benefit the development of the Pill^3 device. There was also no additional cost to download this IDE as it was free for any user, the Arduino system also offers simulation tools that could help to avoid any kind of errors or mistakes that could lead to the damage of some of the components to be used in the Pill^3 device.

The Arduino needs an operating voltage of 5V, it brings a total of 14 Digital pins that would be mostly used for the sensors and motor. The board has an ATmega MCU unit that provides a 48KB flash memory, 6KB of SRAM, a clock speed of 16MHz and it includes a radio module which of the Pill^3 would be an extra feature we might not need but could be further examined for the development of the device as it would not generate any extra cost for the overall development. The Wi-Fi module included in the board was a self-contained SoC with an integrated TCP/IP protocol stack that could provide access to a Wi-Fi network, or it could be even used as an access point. The molecule utilized by the arduino was NINA-W102. An extra feature of this board was that it included a BLE and Bluetooth client and host device which could be easily implemented in the board using the libraries available in the IDE. Overall, it was a board that provides an extensive list of solutions to any developer.

5.6.2 TI LaunchXL - CC3235S

Our second option was the TI Microcontroller specifically the MSP430, these MCU's come from a mixed-signal microcontroller family, which have been built to serve various wireless-enabled applications with a wide variety of systems architecture. These MCU units are designed and built to serve different kinds of connectivity architectures. For the wireless connectivity TI has created a subsection from the MSP430, this subsection deals mainly with wireless communication between the board and any kind of external device, this group was known as the TI CC430, this boards provides to the developer a tight integration between the MCU core, software, and peripherals, other features are added to this but these could be considered the most relevance since these features allow the user to have a better understanding and usage of the board. The board that would be considered to be utilized by TI would be the LAUNCHXL-CC3235S.

The board LAUNCHXL-CC3235S was a dual band launchpad development kit, the board focuses mainly in the single-chip security solution and wireless microcontroller unit which it would integrate the dedicated applications of the ARM cortex-M4 MCU and it provides to the developer a processor that could run 2.4GHz and 5GHz Wi-Fi along with a variety of security protocols. This board features an on-board emulation and sensors for a better development, additionally the board could be connected directly to a PC to program it using the TI IDE known as Code Composer Studio. The board alone offers about 2x20 pin stackable connectors for the developer to use.

The LAUNCHXL-CC3235S board could support the 802.11 a/b/g/n protocols which for the Pill^3 would be sufficient, the advantage of this was that the Wi-Fi was embedded in the bord which like it was mentioned in previous cases this was a key to the Pill^3 design as it would be able to cut some of the cost if the overall design. As we know TI has been one of the companies that includes in their boards security protocols, which was in the actual chip of the board. This would be beneficial to the Pill^3 as it wouldn't be as obvious that the board includes any kind of external chop that was specifically designated for the Encryption authentication of the device. Compared to other boards, the TI offers an inner encryption system that is in a single ship. Similarly to the Arduino, the LAUNCHXL-CC3235S supports a low energy coexistence Bluetooth, which gives the Pill^3 more options for the connection between the user and the device.

One of the advantages of using the TI products was that there was a lot of documentation that could be found about the board itself. Ti has been one of the few companies that was able to provide an extensive and detailed user guide, this guides goe over the pin layout, possible application, limitation of the board, available libraries on the IDE as well as guidelines that need to be followed for the SimpleLink Wi-Fi options as well as the IoT solution layouts. Lastly, TI also offers a hardware guide that was very specific to the board we selected, in this case the LAUNCHXL-CC3235S.

The advantages of this board was that we have been exposed to Code Composer Studio in previous classes, giving us familiarity with the IDE. This IDE uses C programming which makes the development of the product much simpler since we would be utilizing a high level language, the advantage of high level languages was that they have wider options for the device improvement in the long run, when researching for MCU unit we decided on looking for boards that are able to be programmed using this kinds of IDE’s. Although we would have to get more exposure with the board alone since it would be the first time we even use this kind of device, all the features indicated on its technical documentation would be beneficial for the development of the Pill^3. Compared to the Arduino the TI microcontroller provides more pins and a better connection with external devices the main thing would be to test the Wi-Fi capability of the board as it it's crucial for the Pill^3 to have this kind of features, since then again , the device alone depends mainly on the notification system that would be linked with the application created the only downside of this board was that the IDE needed was not as user-friendly as the Arduino, since the Arduino uses more simple instruction set compared to the TI board which would be more focus on the embedded side of programming.

5.6.3 TESSEL 2

Third we have the TESSEL 2. TESSEL 2 is a robust IoT and robotics development platform which leverages all the available libraries of the Node.Js in order to create various devices. This development board has on-boad WiFi capabilities that allows the developer to build scripts in Node.js, this board provides a connected hardware prototype system that has a wide range of applications, this board was part of the AVR MCU unit. For the purpose of our device we would be considering working with the Tassel 2.

This board also gives the developer a very simple software environment that allows the developer to set the WiFi credentials and easily manageable authentications needed based on the device being developed. The advantage of this board was that while it might not have a specified IDE, you could program the board using JavaScript which was a much simpler programming language to use, Javascript was also considered a high level language, which was an important feature to be considered for the development of the Pill^3 . This board comes with a 10 pin module port to add sensors or any other kind of external hardware that needs to be connected for the application being developed, two USB ports, a 10/100 ethernet port, and a microUSB connector for power and tethered programming. This board also offers a 802.11/b/g/n WiFi setup, a 580MHz Mediatek router-on-a-chip, a 48MHz SAMD21 coprocessor, and 64MB of DDR2 RAM with 32MB of flash. This board also provides an Ethernet wired connection, which could be very useful during the development of the board, however, it was more important that the Wi-Fi capabilities were properly functioning.

The disadvantage of this board was that there was not enough documentation for the possible applications nor the board structure itself. Even though the board feature matched with the criteria of the Pill^3 device it would be rather risky to use a board that was not well known, as there could be any potential errors that could arise from the development of the Pill^3 device. Another downside of this board was that there was not a security encryption system nor chip. When researching for MCU units it was extremely important to have this encryption system or chip as we would be working with sensitive information, duie to this we came to the conclusion that the TESSEL 2 board might not be a good option for the Pill^3 device.

With all these features in mind we could say that the TASSEL 2 could be a good option for the development of Pill^3 because it offers a fast and simple developing environment. The only downside of this board was that the majority of the team members are not familiar with the board nor their software implementation, even if it uses JavaScript which was not as complicated as the TI software implementation, it could be hard to debug and run the code each time since we would be analyzing any sort of error from the user computer command prompt, using an IDE would be ideal for the development of the Pill^3. That was in terms of the software, if we take into consideration the hardware it would not be a good option since we would be limited by the amount of pins, at this stages of the project it was still unknown how many pins or cables we could be needing for the development of the product, however, it would be good to have extra amount of pins assuming multiple sensors and LCDs are connected to the board.

5.6.4 PIC - IOT WG AC164164

Fourth we have the PIC microcontroller-IoT microcontroller. This board was an option we considered quite a bit when researching for microcontrollers. This board gives us the flexibility to create any sort of IoT device, ranging from intelligent lighting to wireless sensor notes. The advantages of this board was that it had a fully certified Wi-Fi module to provide the developers a simple form to connect embedded applications to the Google Cloud IoT platform. The advantages of this platform was that it provides the user a fully managed service that allows the user to connect and manage any data sent to this cloud.

The PIC -IoT WG board has the microchip ATWINC1510 module embedded in the board, a single band 2.4GHz network controller that was meant to better enhance some of the low-power IoT applications. Furthermore, the whole board has very low-power consumption which was beneficial to the Pill^3 device. The PIC - IoT board has 8Mb of on board flash memory which meets the minimal criteria for the development of the device, here the ATWINC1510 module would offload all assigned networking tasks give from the main CPU while at the same time was able to automatically provide a secure socket connection and the server authentication to the Google Cloud. The slight disadvantage of this was that the board was connected to a specific cloud service which could be more expensive compared to others, with that in mind, for the Pill^3 having that flexibility was very important since we could potentially reduce costs if we select a low cost Cloud Service.

Because the Pill^3 would be working with sensitive patient information, a CryptoAuthenticator device would be needed. The PIC - IoT board has this kind of authenticator embedded in the board, the one that it specifically utilizes was the ATECCC608A, which provides a trusted and protected identity for each of the devices that uses this board and could be safely authenticated. Even though it's something that could be later added either on the application or the program it was less inexpensive to have it embedded in the board.

The PIC - IoT board was designed from the ground-up by Microchip technologies; the advantage to this was that the board came with its own IDE (Integrated Development Environment) and a Code Configurator as well as a Rapid Prototyping tool. Having an IDE for the Pill^3 development was very important since we could easily detect any problems that might arise as we are building the code, not only that, but it provides us with a robust library to work with as we are developing the program, in our case the implementation might not need as many libraries but a variety of functions could be added due to the amount of components to be used for the Pill^3 device.

This board comes with a PIC24FJ56 16-bit Microcontrolller, with a 128KB flash memory and 16KB of SRAM. It has the WINC1510 Wi-Fi Smart Connect IoT Module with a single spatial stream in the 2.4GHz ISM band. It supports the IEEE 802.11 WEP, WPA and WPA2 security. It was able to support a full networking stack. UART, SPI peripherals are included in the board, as well as 8Mb of Flash memory. It includes a 28-pin package. For the CryptoAuthenticator, it operated with a 1 mA current, was able to protect any storage for 16 different keys, and was able to generate an Ephemeral key and an authenticator key in order to prevent any on-board attack.

The board includes a Digital Temperature sensor as well as an ambient light sensor, these two sensors are helpful for the development of our device. Since the sensors are already embedded in the board this could help to reduce the cost of the overall device as there would be no need to add any extra temperature sensors. In Addition to this, the board includes a voltage regulator circuit as well as a Battery Charge Management Controller.

5.6.5 Arduino Nano IoT 33

Fifth we have the Arduino Nano IoT 33 microcontroller. This board was an option we considered towards the end of the design. This was a suggestion made to us since it's a board that has more pins to offer and has a lot of similarities to the Arduino Uno, and has the wifi connection we need for the Pill^3. The advantages of this board was that it had a fully certified Wi-Fi module to provide the developers a simple form to connect embedded applications to the Google Cloud IoT platform, added to this we have the flexibility to use bluetooth to connect to the MCU which could facilitate the testing process. This board gives the user and the developer the flexibility to connect to the Firebase in which all data given by the user will be later stored and used by the MCU.

The Arduino Nano IoT 33 board has the microchip SAMD21 Cortex®-M0+ 32bit low power ARM MCU module embedded in the board, a single band 2.4GHz network controller that was meant to better enhance some of the low-power IoT applications. Furthermore, the whole board has very low-power consumption which was beneficial to the Pill^3 device. This chip is a single 32-bit AMBA-3 AHB-Lite system interface that provides connections to peripherals and all system memory, which includes Flash and RAM, added to this the I/O port bus interfaces to the PORT with only 1- cycle loads and stores.

Because the Pill^3 would be working with sensitive patient information, a CryptoAuthenticator device would be needed. The Arduino Nano IoT 33 board has this kind of authenticator embedded in the board, secure communication is ensured through the Microchip® ECC608 crypto chip. Even though it's something that could be later added either on the application or the program it was less inexpensive to have ot embedded in the board, and luckily the Arduino Nano IoT 33 provinces that.

The Arduino Nano IoT 33 board was designed from the ground-up by Arduino technologies; the advantage to this was that the board came with its own IDE (Integrated Development Environment) which makes the editing and testing process even faster. Having an IDE for the Pill^3 development was very important since we could easily detect any problems that might arise as we are building the code, not only that, but it provides us with a robust library to work with as we are developing the program, in our case the implementation might not need as many libraries but a variety of functions could be added due to the amount of components to be used for the Pill^3 device.

This board comes with a SAMD21 Cortex®-M0+ 32bit low power ARM Microcontroller, with a 256KB flash memory and 32KB of SRAM. It has the MKR WiFi 1000 Module. It supports the IEEE 802.11 WEP, WPA and WPA2 security. It was able to support a full networking stack. UART, SPI peripherals are included in the board.

Table 10: MCU Comparison

Features	**Arduino UNO Wi-Fi REV2**	**LAUNCHXL-CC3235S**	**TASSEL 2**	**PIC - IoT WG board**	**Arduino Nano IoT 33**
Digital I/O Pins	20	40	20	5	14
SRAM	6, 144 Bytes	262,144 Bytes	67,108,864 Bytes	16KB	32 KB
Flash Memory	49,152 Bytes	1,048,576 Bytes	33,554,432 Bytes	128,000 Bytes	256,000 Bytes
Operating Voltage	5 V	5 V	5 V	3.3V	3.3 V
Wi-Fi	YES	YES	YES	YES	YES
Encryption	YES	YES	NO	YES	YES
Clock Speed	16 MHz	5 GHz	580 MHz	16 GHz	48MHz
Cost	$44.80	$44.99	$44.45	$35.99	$18.40

The table above reflects the difference between boards and the key factors to be considered for the design of the product. Looking at each of these boards we could see that the PIC-IoT board has a faster clock speed, however, it has a enormous limitation with the pins compared to other boards, even though it could function with as little as 3.3V it has a lot of physical limitations that would not benefit the development of the Pill^3 device, because of this we decided on not to work with this board. Next we have the TESSEL 2 board, this board was one of the main options we had for the Pill^3 however, after further inspection the board does not provide a secure encryption system which would be a vulnerability of the Pill^3 device, the application would be connecting directly the device alone but because we are connecting the Pill^3 with an external device that could potentially caused a security threat if we so decide to go for the TESSEL 2 that has no security encryption embedded in the board. Followed by this was the TI microcontroller, this board was one of the options we were highly considering but after further consideration we noticed that the cloud service we decided to use would not work with the TI board, other than that this MCU unit met all the needed criteria for the Pill^3. On the first design used the Arduino Rev 2 board since it meet all the criteria for the Pill^3 and gave us the flexibility to add more for future development, there were features on it that were not needed for the early states of the device but could be potentially used in the future, in fact, this board gives us the option to interface with other components without worrying about compatibility. After testing the board with the other components we noticed that the MCU did not have enough digital and analog pins forcing us to use a multiplexer. Because of this, we decided to switch over to the Arduino Nano IoT 33 board since it was a cheaper option and it met the minimum requirements for the Pill^3.

This board offered more pins and a better connection with the Firebase that was needed in order to store user data.

5.7 Indicators

The primary indicator for our project was the notification system from the application implemented, this system would notify the user for any refilling needed, if the pill container was empty, times the user need to take their medication based on the schedule inputted from the user, it would let the user know if any mediation was not taken and if they would like to reschedule. This notification would come from the user's device whether it was the phone or even tablet. Another indicator to be considered for the device was an LED, which could be implemented in different ways, mainly in the device, these LEDs could be needing more than one bulb. And lastly, we have the speakers which could be considered an indicator for this device

5.7.1 LED

For the Pill^3, individual or various LED bulbs could be used, these bulbs could be an indicator to the user when the pill has been dispensed, these bulbs would be on the exterior of the device. There could be an LED bulb added inside of the device for the developers to know if there was an error or if a pill was stuck in the process of being dispensed. The LEDs would be on the cheaper end in terms of actual hardware since most of these bulbs could be found anywhere and at a reasonable price.

An advantage of these LEDs was that they are because they plastic shell and it's very unlikely for them to break, not only that, but these LED could be controlled directly from the microcontroller and a DC power source, which makes the programming of these lights easy regardless of the board or chipset used, these lights don't require much power either. The one thing to have in mind was that if these LEDs are left on for a long period of time the bulb could burn out making it difficult to determine the life of the LED.

5.7.2 Speaker

Another indicator for the Pill^3 would be a built-in speaker that would function in parallel with the application, the idea behind it was that we have multiple ways to let the user know when the pills have been dispensed or if it needs to be refilled. Even if the user were to get multiple notifications from various sources, the goal behind it was that the consumer does not forget to take their medication.

This speaker would function like a clock alarm, along with the LED and the notification system from the app. The speaker alone was also inexpensive, there are many vendors that could provide a cheap speaker, moreover, to program it would not be complicated either since it was a simple hardware device that could communicate easily to the microcontroller. On the other hand, we need to be cautious with the speaker mentioned since it could be powered using high voltage, because it could cause a short circuit, ideally we use a speaker that has a clear sound and was easy to implement.

5.7.3 Application

The last indicator we have for the device was the notification system, this would be done through an app that could be downloaded into the user devices. The user has the option of having multiple devices connected with their pill dispenser; the idea behind this was that if the user were to forget to take a scheduled medication it would let an authorized user know about it. The notification needs to be on on the consumer device at all times since the Pill^3 would be communicating with the user through it.

The benefits of this system was that there was no hardware cost behind it, however, there could be a cost from the cloud services where we would be storing the consumers information. There are many cloud services that could be used for this product, but at this point of the design such systems are yet to be decided. In summary the most effective indicator for our device would be the notification system that would be then again coming from the application implemented for the product.

6.0 Related Standards and Realistic Design Constraints

The purpose of this section was to discuss professional standards and design constraints that relate to this project. Due to the design specifications of this project, FDA medical hardware guidelines must be followed in addition to IEEE standards. The construction and implementation of this device follows all guidelines as written and abides by all legal requirements set forth by these associations.

6.1 Standards

All standards for this project were discussed and decided upon by the team in the early stages of our project, as they would impact the design process and methods that would be used throughout the research and construction of the Pill^3. This section of the documentation examines each standard that was deemed relevant to the project. Each standard was described, the reason for its inclusion was listed, and the impact it has on the design moving forward was discussed.

6.1.1 IEEE Standards

As this project involves the design, construction, and usage of electrical components, standards and regulations put forward by the Institute of Electrical and Electronic Engineers (IEEE) may apply to the Pill^3. These standards exist as a way to enhance efficiency and safety when designing new products, and to confer confidence to the consumer of a product's reliability. IEEE standards pertain to both industry standards for electrical connections and wireless communication, as well as how to document and discuss tool usage. The team decided that several IEEE standards pertained to the Pill^3. The standards chosen would be described and the reasons for their inclusion discussed.

IEEE 1588-2019 Standard for a Precision Clock Synchronization Protocol: This IEEE standard defines a way to precisely synchronize an internal electric clock through a networked system. This particular standard supports “synchronization accuracy and precision in the sub-microsecond range with minimal network and local computing resources”, meaning time could be kept accurately through a Wi-Fi connection. The Pill^3 needs accurate measurements of time to notify the user and dispense medication. This standard would help to fulfill this requirement, and since the Pill^3 would already be using Wi-Fi to communicate with the smartphone application the clock could be synchronized through this connection as well.

IEEE 82079-1-2019 International Standard for Preparation of Information for Use: This IEEE standard refers to the inclusion of instructions for use of a commercial product and the information required in that document. The standard states that information for use could be described as “necessary for the safe use of a product; helpful for the efficient and effective use of a product; and often necessary to fulfill market, legal, and regulatory obligations.” This information must apply to all phases of a product life cycle from assembly to disposal, specify which parts of maintenance must be performed by a professional and provide general knowledge on how to operate the product. As the Pill^3 was intended for commercial use, this standard applies to the project, and informative instructions for use must be included with the product.

IEEE P11073-10472 Health Informatics -- Personal Health Device Communication: The purpose of this IEEE standard was to establish an industry accepted definition of the communication of medical devices, specifically monitoring and managing devices. It defines medication monitoring devices as “devices that have the ability to determine and communicate (to a manager) measures of a user's adherence to a medication regime.” In the context of this project, the monitoring device was the physical body of the Pill^3 and the manager was the smartphone application that tracks distribution times and pill supplies. As such, it was decided that this standard pertained to the project and the way communication was handled between the device and the application.

IEEE 829-2008 Standard for Software and System Test Documentation: This IEEE standard establishes a testing and documentation procedure for both the physical hardware of a device and the software associated with it. Our team decided to consider this standard when testing and developing the smartphone application, as it makes the process more streamlined and could help with the detection and correction of errors.

IEEE 1149.1-2013 Standard for Test Access Port and Boundary-Scan Architecture: The methods described in this IEEE standard pertain to the testing of Printed Circuit Board (PCB) components. As a PCB designed specifically for the Pill^3 would be used in this project, a standardized way of testing and confirming the intended functionality of the board was required. This standard includes “A set of test features was defined, including a boundary-scan register, such that the component was able to respond to a minimum set of instructions designed to assist with testing of assembled printed circuit boards.” These procedures would be used to ensure that the electrical components connected to the PCB, and the PCB itself, work properly when assembled in the device.

6.1.2 FDA Standards

As this project was intended to be used as a medical device in the United States of America, our team decided that the Food and Drug Administration (FDA) standards pertaining to the manufacturing, sale, and usage of such devices would apply to the Pill^3. This organization oversees the medical industry in America, and as such has various standards in place to protect consumers and suppliers alike. The standards that were deemed relevant to the Pill^3 are explained and discussed below.

Medical Device Reporting - 21 CFR Part 803: This standard imposed by the FDA mandates that any occurrence where a medical device might have “caused or contributed to a death or serious injury” must be reported to the FDA. The purpose of this standard was to ensure that any life threatening defects of any medical equipment are caught quickly and remedied efficiently. While the Pill^3 should not pose any threat to the consumer, this standard was important in ensuring that any unforeseen issues that may arise from the construction of the device are promptly fixed. As such, our team decided that this would be an important standard to consider when designing the device.

Establishment Registration and Medical Device Listing - 21 CFR Part 807: As part of the guidelines put forward by the FDA’s Center for Devices and Radiological Health (CDRH), the division of the FDA that oversees the manufacture and sale of medical devices in the United

States, any manufacturer or distributor of medical devices both foreign and domestic must be registered and approved by the FDA before any sale was allowed to take place. Also included in this standard was the device listing guidelines, where it states that any medical device must also be registered with the FDA. While these standards do not directly affect our team's effort to create this device, it would be vitally important to any further manufacturing or marketing done by our team after initial development. As such, it was decided that these would be relevant standards to adhere to.

Labeling - 21 CFR Part 801: This standard, set in place by the CDRH, relates to any informational labels and literature included with the device. It details the type of information that should appear on informational labels, how and when symbolic labels are to be used, and what the contents of the user guide literature should contain. Although the prototype and the presentation version of the Pill^3 do not require and would not include informational labels such as these, any production further would require them to be present. For this reason, our team decided to include these standards when considering the design of the device. Important informational labels would be required for this device to be marketed fully and are thus an important standard to be mindful of.

Performance Standards For Electronic Products - General - 21 CFR Part 1010: This section of the CDRH's set of standards for medical devices details the general standards used for any medical device containing electrical components. It includes a definition of the scope of the FDA's authority to monitor and control electrical components, steps for certification and identification of any medical device containing such components, guidelines concerning any variations of a device that may arise from further research, and testing procedure for these types of devices. As the Pill^3 contains several types of electrical components, this set of standards was referred to heavily to ensure that there were no issues with the choices for any piece.

6.1.3 Safety Standards

To ensure that no accidents occurred during the production of the Pill^3, our team followed several safety guidelines put forward by various accredited organizations. These standards included information about safe work environments, power tool use, safe construction methods, and product quality factors. Through proper use and implementation, our team had no injuries related to the design or construction of the Pill^3 and created a product that would pass any fire and health inspection it was subjected to. The standards followed are listed and discussed below.

NFPA 70E Standards For Electrical Safety in the Workplace: This series of guidelines was developed by the National Fire protection Association (NFPA) at the request of the Occupational Safety and Health Administration (OSHA) and details the safe use of electricity in a work environment. This was to ensure that no workers present suffer any form of bodily harm and that there are no electrical accidents which may lead to death. Our team observed these standards both while using electrical equipment to build our project and while designing the Pill^3's electrical layout. These standards were followed so that no member of the team was harmed and so that there was no risk of the Pill^3 being a fire hazard.

OSHA 1910.137 - Electrical Protective Equipment: This standard, part of a subpart of the personal protective equipment standard, contains information pertaining to the types, usage, marking, and storage of electrical protective equipment. It lists the proper environmental safety precautions needed to work on electrical systems and how to set them up. As our device needs the electrical components and circuitry to be assembled by the team members, it was decided that proper knowledge of the safety procedures put forth by OSHA was needed. As such, this standard was referenced while obtaining and setting up safety equipment to ensure that the electrical testing was conducted smoothly.

OSHA 1910.304 - Wiring Design and Protection: This OSHA standard within the electrical subcategory details the proper steps for identifying a ground, grounding the circuit, managing branch circuits, and other safety procedures to be observed while working with circuits. While this information was especially important for larger systems with higher voltages or currents, our team decided that the principles should still apply to the circuit used within the Pill^3, and that the standard should be observed during circuit design and construction. As such, the circuit designed for and used in the device meets with this OSHA standard, and has it’s electrical ground properly labeled and protected.

6.1.4 Miscellaneous Standards

Several additional standards were deemed as necessary but did not fall under either the FDA or the IEEE organizations and were not considered safety standards. These standards include general industry standards and standards that apply to the United States as a whole. Any standard listed was followed procedurally and correctly, leading to a product that was both improved upon and ready to be retailed in the United States.

Standard Voltage - Department of Energy - Office of Electricity: Within the territory of the United States of America, typical wall outlets are rated for 110 to 120 volts at 60 Hertz. This standard was somewhat arbitrarily decided when electricity was first becoming widespread in the U.S. If our team were to market this device beyond the United States, each country's individual standard would need to be accounted for. To make the Pill^3 a reasonable and reliable product for use in the United States, this input voltage standard needs to be observed. As this device was intended for wired use within American households, our team decided that this standard needed to be followed.

6.2 Realistic Design Constraints

This section would discuss several design constraints that greatly affect the composition and construction of this project. To realistically complete this project, the previously listed specifications as well as several constraints not within the original scope of idea planning would need to be met. Of these, economic and time constraints would be the most impactful, as this project seeks to be more affordable than other devices currently available that deliver a similar service. Also of notable importance are environmental, health, and safety constraints, as this device seeks to be used as a personal medical aid and thus must not hinder the user's health in any perceivable way.

6.2.1 Economic Constraints

Cost of production for the Pill^3 was perhaps its biggest constraint, as this project has no financial sponsor and seeks to be affordable to consumers. As such, this project aims for a total production cost of less than $200 per unit. While this does not include research and development costs, the goal was to keep total expenditures below $300. All expenses are to be covered by the team and would be evenly split among the 4 members, leading to a maximum contribution of $75 per member. This amount would need to cover the cost of the microcontroller, motors, LCD display, speakers, mechanical components, and housing unit materials.

Efficient design and quality, low cost components would need to be thoroughly researched and implemented to conform to this limit. The team would also determine the most suitable housing material and construction method to minimize cost while ensuring device reliability. In a similar vein, housing materials would be utilized sparingly until a final product design was decided upon. To facilitate keeping costs low, prototype mechanisms would be constructed and tested using components already owned by team members, and only once a mechanism was fully tested would the corresponding part be purchased.

In regards to the consumer price the team seeks to retail this device for no more than a $200 markup of the production cost, resulting in a Manufacturer's Suggested Retail Price (MSRP) of $400 or less. This price point would allow the Pill^3 to remain as the affordable option in the market while still recouping research, development, and manufacturing expenditures. Compared to a leading competitor like Hero which promotes a "Pill Dispenser" with a low cost, our device may initially be more expensive but the lack of a $30 monthly paid subscription for device service and app access would make our device the more affordable option in the long term. Additionally, when compared to the Medacube, another device used for a similar application but with a $1800 retail price, our device was not only significantly more affordable but also includes a smartphone application for setup and reminders.

6.2.2 Time Constraints

Due to the nature of this project and the Senior Design course structure, there are only nine months allotted to research, design, prototyping, and construction of the device. In an attempt to alleviate some of the strain associated with such a short timeline, our team began meetings two months ahead of time during the summer semester. These meetings were used to create an early timeline and discuss possible project ideas so that our team could immediately begin working towards project deadlines when the course started.

The entirety of Senior Design 1 would be dedicated towards researching and designing this device, as well as process documentation. Initial design research, as well as market competitor research, would be done during the first month and would create the basis for future design ideas and checkpoint planning. The next two months would be spent investigating the standards that apply to this device, as well as identifying the electrical components that would be used to make the device operate. The month of November would be dedicated to writing a large portion of the report and writing the initial version of the operating program. The companion app for the Pill^3 would also begin to be developed this month. To ensure that all deadlines for Senior Design 2 are met, the printed circuit board, microcontroller, motors, LCD display, speakers, and other electrical components would be chosen, ordered, and tested during Senior Design 1. Due to the tight time constraints, our group plans to continue meeting weekly throughout winter break, and would begin construction of the prototype design. To ensure that all deadlines for Senior Design 2 are met, the printed circuit board, microcontroller, motors, LCD display, speakers, and other electrical components would be chosen, ordered, and tested during Senior Design 1.

The entirety of Senior Design 2 would consist of construction of and testing of the Pill^3 and preparation for the final presentations. The initial prototype would be assembled within the first two months of Senior Design 2 to allow ample time for solving any issues that may arise from the design. The companion application would be finished during this time as well and would be tested in parallel with the initial prototype. Two group members would be fully dedicated to the physical device, one member would be solely working on fine tuning the application, and the final group member would work on the interfacing between the program and the microcontroller connected electrical components. With this division of work, prototype testing and program testing could occur simultaneously, and any updates to either would be seamlessly integrated into the project. After testing and fine tuning was completed, the remainder of Senior Design 2 would be dedicated to perfecting the documentation of the project and preparing for the end of semester presentations.

6.2.3 Environmental Constraints

While this device was not intended to have any direct effect on the overall environment, nothing built using electrical components was completely free of waste during manufacturing. Due to the nature of the component materials, discarding the device could also cause damage to certain ecosystems. Additionally, the Pill^3 may have another unintended effect on the environment via reduction of prescription drugs being discarded. As such, the environmental constraints of the manufacturing, usage, and disposal of the Pill^3 must be discussed.

During construction of the device, silicon and various types of trace metals are processed into electrical components like sensors, microcontrollers, motors, and wires. The mining, transportation, refining, and fabrication processes for all of these materials produce large amounts of carbon pollution, and some release toxic chemicals into the atmosphere or disrupt local environments. Mining was especially destructive to the environments directly around the area of work. Gold mining has been known to pollute water supplies if handled incorrectly, and sifting for silica, a key component in the fabrication of silicon, could cause land deterioration and loss of biodiversity. As such, our team would only source components made from responsibly sourced raw materials in an attempt to minimize the impact the construction of this device has on the environment. Another large source of pollution in the fabrication process was the carbon emissions generated during the transportation of raw materials and the finished products. While there isn’t much our team could do to reduce the amount of carbon produced from a shipping container, the footprint of the Pill^3 could be minimized by making as few shipments as possible and ordering or shipping in bulk.

Reduction in consumer medical waste may occur through the intended use of the Pill^3 due to the nature of the device. The World Health Organization (WHO) classifies “expired, unused and contaminated drugs and vaccines" as among the leading types of medical waste, contributing to toxic and dangerous environments. By ensuring that the consumer does not forget to take their prescribed medication and by administering it in a timely manner, the number of pills that expire or are discarded by consumers would drop significantly. This would positively affect the environment, as medical waste could cause drastic changes in the chemical balance of water sources and the organisms that live there. Additionally, a reduction in this type of medical waste would increase the efficiency of prescription medicine usage, meaning that less pills would need to be manufactured to make up for the discarded ones and resulting in less pollution from the manufacturing processes.

6.2.4 Social and Ethical Constraints

By the nature of the design of this device, the Pill^3 was designed to assist consumers and patients in taking prescribed medication or supplemental vitamin pills. As such, the goal of the project was inevitably to help people by making their life easier, if only slightly. When our team decided to make a medical device, one of the initial core ideas was "a project people could rely on". To this end the Pill^3 was designed with the Hippocratic Oath of "do no harm" in mind. To achieve this, the device had to have both ethically sourced components to minimize its "ethical impact", and to be marketed responsibly. The effects that each of these constraints had on the design and process of the Pill^3 are discussed below.

One major source of contention within the electronics market was the labor used to create electrical components and manufacture electrical devices. To maintain lower cost of production than competitors, some companies would choose to use factories or manufacturers that employ inhumane working conditions, non-livable wages, or drastic worker hour requirements. These types of manufactures are seen as unethical but, while they may be frowned upon by an informed public, they are not illegal. To realize the ideal of an "ethical product" that our team has, all components both electrical and physical were purchased from vendors and manufactures with a history for good, ethical practices. Our team wanted the entire process of the Pill^3 to be ethical, not just the finished product, so each vendor was thoroughly vetted before any orders were placed. This limits the pool of manufactures we could choose from dramatically. This constraint clashes heavily with the budget constraints, as ethical manufactures tend to have higher prices for their components to offset the extra costs of a habitable work environment. Our team had very limited options for both ethical and affordable components, made even more difficult by the fact that certain major shipping companies have recently been exposed for inhumane work environments. Despite this, the Pill^3 was able to be manufactured without any ethical dilemmas from the source components.

Another ethical issue arises from the transportation, sale, and quality of the product once it arrives. Similarly to the manufacturing process, the transportation process was held to the same standard. Only companies with reputable records and ethical practices would be used for shipping to ensure that the entire process from factory to consumer hands was both socially and ethically acceptable. Additionally, any vendor through which this device was to be sold would be held to the same criteria. Finally, the device itself must be of suitable quality. It must fulfill all promises made by the advertisements and informational material included with the device, and it must be both durable and long lasting. This means that the team must design the device to withstand several years of normal use. As such the components and engineering techniques used in the design and manufacturing process need to be of suitable quality. This again places a burden on the financial budget and constraints of this project, while also introducing another layer of quality that needs to be assured to the consumer.

6.2.5 Health and Safety Constraints

As the Pill^3 was a medical device, its construction must take into account how to improve the life of its consumer. Consequently the health and safety of the user and their friends and family were of the highest priority when designing this device, as a medical device that could harm or injure the consumer would both not meet standards put forth by the FDA and the IEEE and would be counterproductive to the original goal of the device. To protect both the main user and their family several features were added to the design. These features came from the constraints our group decided upon; that the device must not pose a threat to small children, that the schedule and amount of dosage must only be editable to the main user, and that no injury may come to any consumer so long as the device was used as intended.

The first constraint pertaining to the protection of children used during the design process was that the device must not be easily activated by children. This was to prevent them from consuming medicine in types or quantities that could be deemed detrimental to their health. The feature that resulted from this constraint was a child lock that required a button to be pressed and held for 3 seconds before the device dispensed any medicine. A child lock such as this would prevent small children from accidentally releasing the medicine. Since small children and toddlers have a tendency to put objects in their mouths to help with sensory motor development, any medicine left unattended by a parent may be unintentionally ingested by that child. The second constraint pertaining to the protection of children was that the device must not be easily tipped and must not slide across surfaces. This was to avoid the device falling onto small children and injuring them. As children of a young age also have a tendency to reach for things on a counter or table above them, our team decided to make the device have a low center of gravity to avoid tipping when grabbed from a low angle. To prevent the device from sliding across counters and off edges, rubber material was used on the bottom of the device to add more traction to the frame. The final constraint was that a child would not be able to easily access the medicine supply. To achieve this, the opening for the pill storage was placed on the top of the device, and simple “child safe” locks were installed on the lids of the containers. This prevents children from opening the top and ingesting the pills in large quantities.

Another area for concern about the safety of the device was the ability to tamper with a set prescription schedule . If a user’s schedule were to be changed or deleted without their knowledge they may fail to take their medicine as prescribed which could lead to sickness or, in certain cases, death. As such our team decided that there should be a layer of security within the app, requiring a password authentication sign in before a user could add, change, or delete a prescription schedule. This would prevent anyone besides the original user from editing their prescriptions. Our team decided that this would be a software constraint after discussing possible situations in which the original user may not have control or oversight of their phone at all times, especially in family situations where young children may use a parent's phone for entertainment. The password authentication would thus be both an additional layer of security for one’s prescription schedule and another kind of “child safety lock” to prevent children from accessing the medicine.

The final constraint our team set for the Pill^3 regarding consumer safety was that it must not pose any danger to the user when operated correctly. This constraint was perhaps the most important and far reaching regarding safety as it meant that every aspect of the Pill^3 design, both electrical and physical, must be thoroughly tested to ensure that no issues could occur that would result in an injury. Aside from the modifications for child safety and password protection, no specific changes were made to the design to accommodate this constraint. Instead every aspect of the design was vetted for possible causes of injury, like sharp corners on the frame or exposed wiring. The design was also checked for any electrical connections that could be fire hazards, and every electrical component was tested before it was installed in the frame to ensure that nothing would break or malfunction. The motors used to dispense the medicine were tested in their housings to ensure that they were not strong enough to harm any fingers or appendages that could get caught in them. Finally, the frame was designed so that there were no jagged or sharp corners that could lacerate or pierce a consumer's skin.

6.2.6 Manufacturability and Sustainability Constraints

Due to the specific niche that the Pill^3 fills, there are several physical design constraints that affect the device's construction and the materials used to manufacture it. These constraints include the size and scale of the device, the materials used to store the medicine, and the air flow system of the electrical components. Each of these issues demands special considerations to the design of the device and how each system interacts with each other. The extra research time and material costs accrued by meeting these specifications are significant enough to merit a discussion of each issue and its solution.

The largest physical constraint of the Pill^3 was its overall size and weight. While it may not seem restrictive enough to be a constraint, since the Pill^3 was intended as a consumer household medical device our team decided that the device’s usability relied upon its ability to seamlessly be integrated into a patient’s home. If the device were too large it would not be able to comfortably fit on most kitchen counters or indoor tables, and would thus not be able to fulfill its intended purpose of making adherining t to scheduled medicines easier. As such the initial constraint for its size was that the device should not exceed 1 cubic foot of space. It was later decided that, due to the amount of medicine that the Pill^3 was intended to store, the dimensions of the device should not exceed a 1x1x2 foot space. The extra height allowed for larger medicine storage cylinders and an increased overall capacity for the device. To further assist with the device being consumer household friendly, the weight of the total design, not including any medicine placed in the storage cylinders inside it, was kept below 20 pounds. This size and weight allow for the Pill^3 to be easily handled and moved, while also having enough of a center of gravity so that it does not tip over easily.

Since the medicine stored in this device was going to be consumed, the material used to store it must meet certain specifications. Most prescription medicines and supplements are supposed to be kept in certain storage environments, so to ensure the integrity of the medicine stored within the Pill^3 the storage cylinders need to be made of materials with similar qualities to those of typical prescription containers. For the qualities of the storage cylinder, the material needs to be FDA approved to be food safe, to be relatively thermally insulative, and to be relatively cheap. The first quality could be met by observing the FDA standards for food and drug storage, and what types of materials are suitable for that. Since most medicine needs to be stored at room temperature, the material chosen for storage must allow for excess amounts of heat to reach the medicine. The material used to make Tupperware containers was a good example of a material that was FDA approved and that would be relatively cheap. As such, a plastic tube or large pipe with similar material properties to the plastic used in Tupperware was chosen to be the storage material for the Pill^3.

Since this device both stores medicine and uses electrical components, proper air flow was essential to ensure that the components do not overheat and the medicine's quality was not compromised by heat. As such, several design constraints were examined when constructing the housing unit of the Pill^3, the most important of which was ventilation. While the electrical components used are small and the amount of heat generated from normal operating usage was minimal, over long periods of time the slight amount of heat may build up and create drastic differences in both temperature and humidity within the Pill^3 storage cylinder. To combat this build up of heat, small slits are carved into the bottom of the device's container, below where the electrical components sit. These slits allow for cool air to flow underneath the Pill^3 and into the device, while simultaneously allowing the warm air out. When the prototype was created, the temperature would be monitored, and a fan could be installed on this opening to promote cool temperatures if the need arises.

7.0 Project Hardware and Software Design Details

This section of the report was dedicated to the discussion and design of the entire Pill^3 project, both hardware and software. Block diagrams for each of these are shown in the Initial Design Architectures, and their application to the overall design are discussed. The rest of this section discusses each subsystem used within the hardware design and every aspect of the programming for both the smartphone application and the actual code for the microcontroller used.

7.1 Initial Design Architectures and Related Diagrams

The block diagram shown was the initial hardware design of our device. As you can see everything starts off in the application on the phone. The app was able to communicate with the controller which then sends the information to the microcontroller via wifi. The microcontroller was the heart of the device as it was where all the information was relayed. For our initial design, it would be powered via an AC/DC converter as it was more reliable than most battery sources. The microcontroller would also be what causes the sensors, LCD screen, motors and speaker to function.

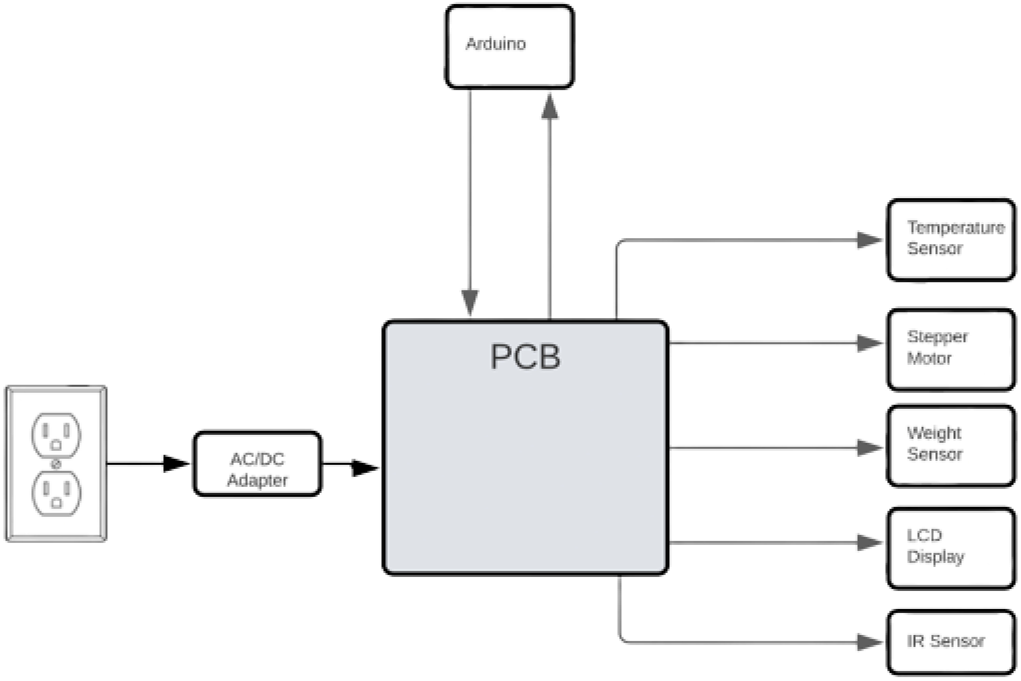


Figure 5 : Hardware Block Diagram

Below was the software block diagram of how our system should function. We begin with having the microcontroller be the link that combines both our hardware and software diagrams. The phone is what sends the information the user sets to a controller module (the controller would be contained in software; it would be where the information gets processed and decides what to send to either the backend or if it has to process something when a user presses a button.

For example, if a user wants to set a time that they want a certain pill to dispense, that was not something that has to be sent directly to the microcontroller, rather it has to be transferred through the backend to the database to store/update the times a user wants a pill time updated.

When the user loads the phone, it would have to request information from the database which would run through the controller module. Once the application boots up, it could request the information necessary, then display that information such as times already set for pills to dispense, what pills at what times have been dispensed, etc.

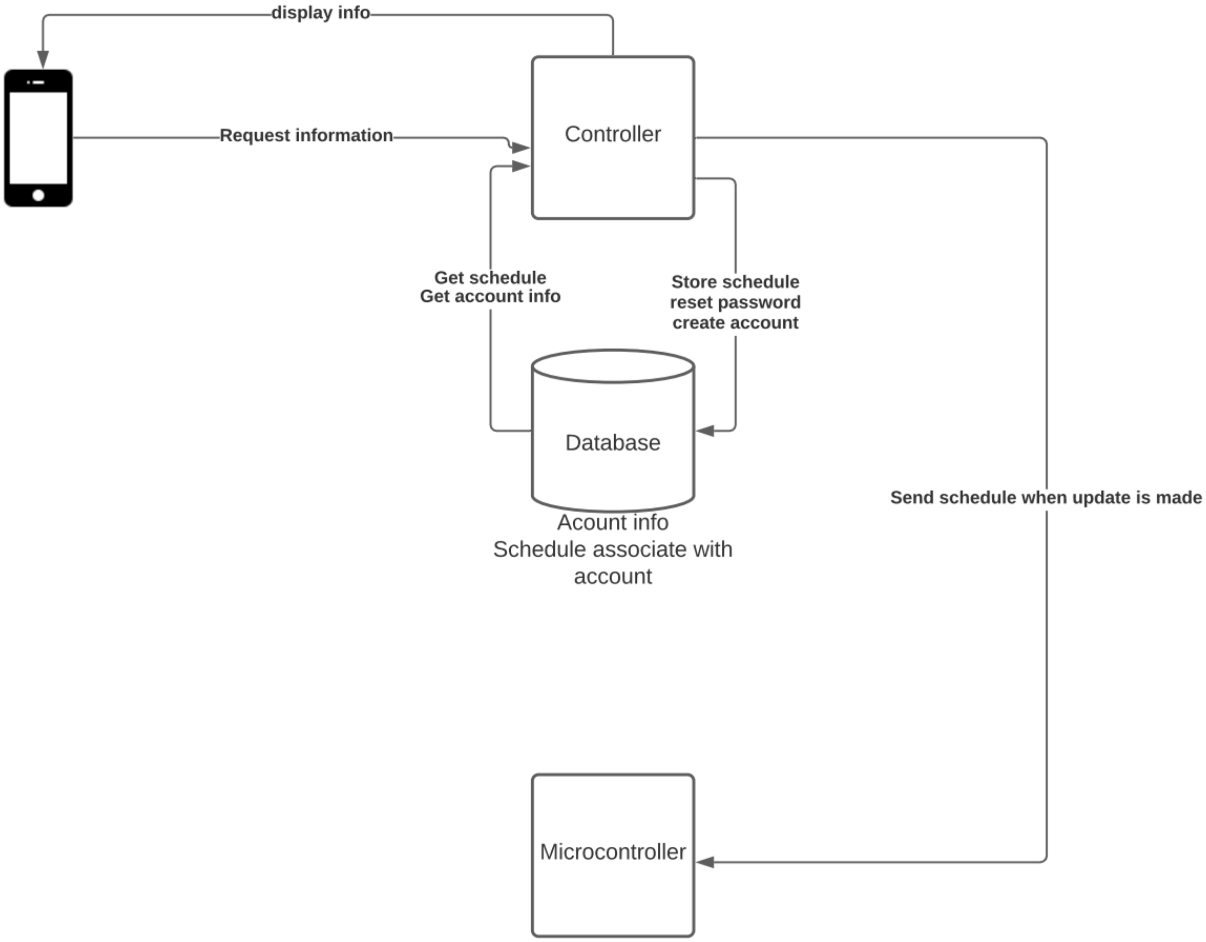


Figure 6 : Software Block Diagram

7.2 First Subsystem: Medicine Storage Cylinder

The Medicine Storage Cylinder (abbreviated MSC) is the vessel in which the consumer places their prescription or supplement to be dispensed. This subsystem contains three main parts; the storage container, the medicine dispersal motor, and the medicine IR sensor. Together, these components could store the medicine, dispense the correct amount, and determine how much medicine was left within the storage chamber.

7.2.1 Construction of Storage Containers

This part of the subsystem was the physical body in which the medicine was stored and distributed. It consists of a disk approximately 25 centimeters in diameter and a cube with a cylindrical hole 1 centimeter larger than the disk cut out of it so that the disk fits snuggly within the cube, but still has room to rotate. The edges of the cube would rise roughly 10 inches above the surface of the disk, creating a bowl shaped cavity within the cube with the disk as the bottom of it. In the center of the bottom of the cube would be a hole for the motor shaft, which would connect to the disk in the cube and the stepper motor at the bottom of the cube. This would allow the disk to be able to rotate independently of the cube. The shape, scale, and configuration of these components are shown in the 3D render below.

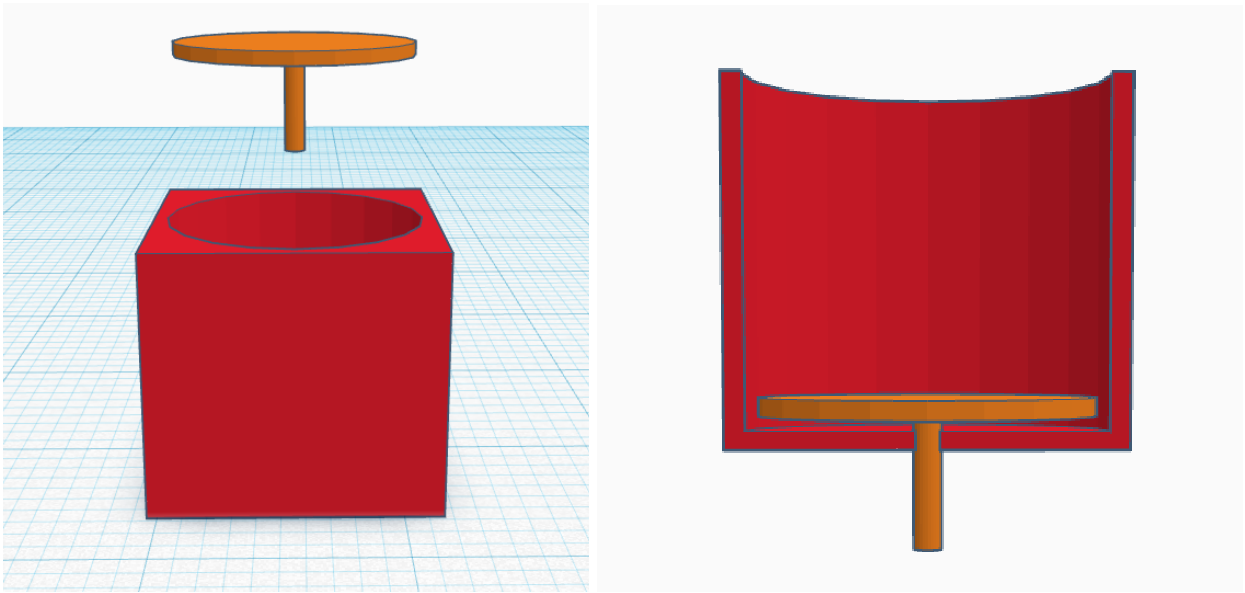


Figure 7: Sorting Disk
(Left) Side view of the container components
(Right) Interior view of the container components

Connected to the cube at a level 0.5 centimeters above the disk would be the feeding tract layer which was a wall of material that was placed slightly above the disk and covers the left half of the disk, separating the disk into the storage side (right side) and the sorting side (left side). At the top of the storage side there would be a curved edge leading to a small opening. This opening connects to a linear path from the storage side of the disk to the sorting side. Once on the sorting side of the feeding tract, the linear path curves towards the bottom of the disk and begins to split into different sized paths, leading to the edge of the disk where the paths cut off at the same point just before the edge. Below is an image of the described sorting path, showing the increasing size of the paths the further they get from the center of the disk. This would help to organize and dispense pills at a constant rate no matter the size.

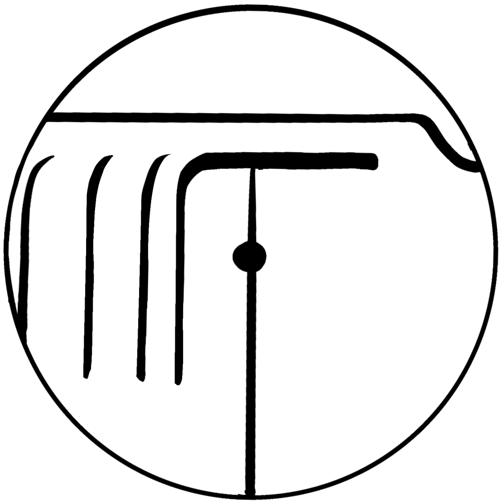


Figure 8: Sketch of feeding tract layer over sorting/storage disk

The materials used for this subsystem would vary based on the use of the component and the amount of contact that component has with the user's medicine. Any component that was in contact with the medicine, including the spinning disk, the walls of the storage cube, and the feeding tract layer must comply with the FDA material guidelines for long term storage of medicine. As such the disk and storage cube would be made of a special FDA approved plastic filament and would be treated before installation into the device. The feeding tract, which would be laser cut from a board of plywood, would be stained and finished in accordance with the FDA guidelines for wooden utensils.

7.2.2 Medicine Dispersal Motor

To dispense the medicine held in the storage container, an electric stepper motor with low speed and high torque was used to rotate the sorting/storage disk. It was attached to the sorting disk by way of a drive shaft that passes through the body of the storage cube, allowing it to rotate along the bottom of the storage container. As the disk spins, the medicine would be pushed up towards the top of the disk and, due to the continuous turning motion of the disk, slide along the linear path in the feeding tract. The configuration of the storage cube and rotation disk with the motor attached was shown below.

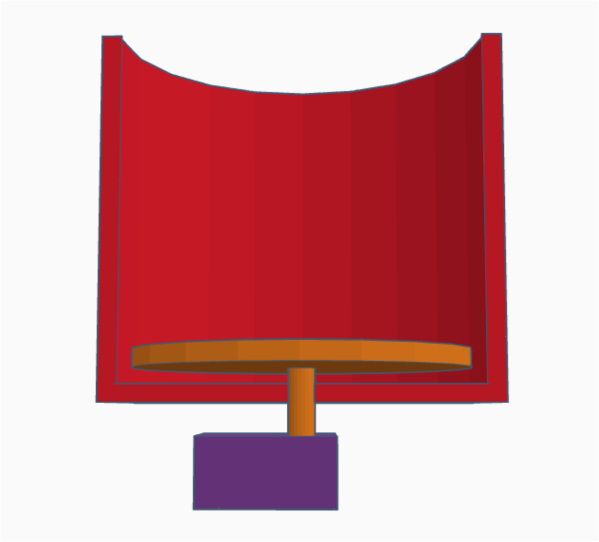


Figure 9: Interior view of the container components with the motor attached to the drive shaft

The best performance motor for this application was a stepper motor as it has both a full 360 degree range of motion and precise control over stop, start, and location. Since this subsystem would use a passive feeding tract to determine a medicines size and space them appropriately, the motor does not need to move very quickly. The device was designed to only allow one pill to be dispensed at a time, so a slower motor allows adequate time between each time the sensor counts a pill as dispensed.

7.2.3 Medicine IR Sensor

To determine the amount of medicine remaining and whether a pill has dropped through the sorting disk, an infrared sensor was used to track pills leaving the feeding tract layer of the disk. The infrared sensor was positioned at the end of the feeding tract, aimed horizontally across the exit of the paths at a strip of black material. While there was no obstruction between the infrared light and the strip of material, the black coloring of the material absorbs the infrared light emitted from the diode of the sensor, meaning that the detector of the sensor does not observe any IR light. Once a pill crossed in between the sensor and the black material, the infrared light was able to reflect from the object and the sensor was able to detect the change in status.

The change in status was used to both accurately count the number of pills distributed (thus estimating the number of pills remaining) and quickly and reliably inform the system of when this change occurred. Due to the nature of the storage device itself, it was difficult to determine the number of pills in the system. To work around this issue, the infrared sensor would count the pills as they're released, subtracting them from the total amount of pills that the user reported they added to the device. This would allow the system to generate a fairly accurate estimation of the remaining number of pills. The team plans to tune the system so that the number of pills remaining was precise within an error margin of 5%. The second important quality of the infrared sensor's position was its role in stopping the rotating disk. Once a pill was counted, the device would be able to receive the information from the infrared sensor and send the command to stop the stepper motor within 0.5 seconds, to prevent any unwanted extra pills from being distributed. Any additional pills that get through must be detected by the system and report the error to the user via the smartphone application or the native display on the device.

7.3 Second Subsystems: User Notification

The User Notification System (UNS) was responsible for alerting and informing patients of the status of the Pill^3 through a combination of audio cues, visual cues, and smartphone notifications. This system would handle communicating medicine dispersal alerts, low supply warnings, and general system health updates directly to the user. By utilizing a low power speaker and LCD display, the device itself would be able to handle this task.

7.3.1 Auditory Alerts

The auditory notification would be handled by a simple low power speaker on the side of the Pill^3. This speaker would be secured behind a mesh to protect it from damage while still allowing the sound to pass through without distorting the message. This speaker would be used to alert a user that the medicine was about to be dispersed, according to the predetermined schedule. The notification sound would consist of a short resonating bell sound that a user would be able to hear from up to 30 feet away, assuming “normal” amounts of background noise. The volume and tone of the alert may also be changed to better suit the environment the user lives in.

7.3.2 Visual Alerts

The visual notification component of the UNS consists of a small LCD screen mounted to the front face of the Pill^3. This screen would be used for both medicine dispersal notifications and for general system information. When notifying a user that medicine was about to be dispersed as per the predetermined schedule, the LCD screen would turn white and flash an alert using the set name of the schedule. When not alerting a user to imminent dispersal, the LCD screen displays the current amount of medicine left in the MSC and the date and time of the next scheduled dispersal. This general information screen would include an additional warning if the user has less than 1 weeks worth of medicine left in the MSC.

7.3.3 Smartphone Application Alerts

While not part of the physical system, aside from the WiFi connector used in the microcontroller, the smartphone application notification feature was included in the UNS as it works in tandem with the visual and auditory alert systems to keep the user informed about their device. When it was time to dispense a scheduled medicine the smartphone application would send a push notification to the device of the user, informing them that it was time to take the prescribed pill. Similarly, this system also notifies the user through their smartphone when there was less than one weeks worth of medicine in the MSC.

7.4 Platform

For Pill^3, there are a few different choices that could be used for displaying, setting times to dispense, along with a few other features that are necessary to make the Pill^3 work efficiently and well. We have the options to use a few different platforms and are listed below. These observations helped us decide which to use through many different considerations.

These platforms are being considered to be used with our User Interface so consumers could have access to

7.4.1 Native Display on Pill^3

With a Native Display, there could be a navigation menu and user interface that would allow users to set times to take pills and medicine, and apply settings to the device. This would also add on an extra cost of buying the display compared to the other options. The other options rely on what the user has; being that most people already have a mobile device, desktop, or internet access for a web application, this wouldn't be using our resources. However, we also need to consider the ease of use that a user would get from not having to manage another device to use Pill^3.

In regards to what would be the framework for the native screen display we have two choices, a touchscreen display with a Raspberry Pi or a touchscreen display using a lower-level microcontroller.

The option to use a Raspberry Pi touchscreen display was an expensive one, but also a common option in the world of native touchscreen displays. Only Python was supported, so everything would be done using Python scripts, which could get complicated as Python was a dynamically-typed language. There was a decent amount of support behind creating graphics with Python; however, it may become similar to using a microcontroller where the work on creating just the graphics would start becoming too difficult which would diminish efficiency.

The other option was to use a touchscreen display with a lower-level microcontroller. This option would require programming in C++, which although fast, would be completely unnecessary for a project of this caliber. It would require more time, more complications, and using it for displays requires a lot of "pixel-perfect" measurements. Using it would require a lot more time to be dumped into just making the screen look presentable, let alone the entire rest of the data collection and processing. In this case, it would be a terrible option because the fast runtime of a microcontroller was too excessive; it would be a waste of time from people on the team when there are better options that could be used more efficiently and have a better final result.

The only upside to using a microcontroller with a native touchscreen display would be not needing internet access, as everything could be stored locally on the device. Although, the negatives of using this option heavily outweigh the positives.

With a native display, we could avoid the need for a backend service as everything would already be grouped together and times and settings could be saved locally to the device. However, there was one big downside that there wouldn't be a way to notify someone if they weren't with the device or out-of-range of the Pill[3]. This was a major downside, as the project relies on notifying people on when to take their medication.

7.4.2 Web Application

A web application would allow for users to log in to a website or application online to access their schedule, make modifications, be notified, and apply settings. There are plenty of frameworks that could be used to make the user interface also look presentable without requiring a ton of effort.

There was plenty of support when it comes to making web applications with a plethora of frameworks such as React, Typescript, Vue, Svelte, or even just using vanilla HTML, CSS, and JS. These options also make working with the backend simpler. Unlike a native touchscreen display, having a web application would require a backend to be able to remotely store information about the user's settings and schedule for certain pills. This would have to be sent over to a microcontroller that could access the internet natively.

With the web application, notifying someone would have to work through the device itself, or else a user would have to keep the web application open all the time, the user's speakers would have to not be turned off, along with a lot of other things that would require the notification system to work. An alternative was to send out texts when a time has reached to take medication which could work in theory, but in reality there was a chance getting constant texts throughout the day may be annoying. This also would require more funding through backend processing, as sending out texts costs money. With a user in control of this (especially if on a larger scale), the amount of texts that would have to be funded was completely in the hands of the customer.

7.4.3 Desktop Application

A desktop application would fill a lot of the same shoes as a web application. Users would open a desktop application to access their schedule, make modifications, be notified, and apply settings. Desktop applications are modular and there are frameworks such as Electron that allow web applications to function as desktop applications. .NET Framework was another popular choice; however, .NET only builds for Windows which would be a downside.

An upside to desktop applications over web applications was that it could run in the background and notify a user while working or doing anything on the computer without having to keep a web browser open at all times. If someone was out, though, they would not be notified which could also be a big issue.

Just as web applications, there is a ton of support when it comes to making desktop applications. Another thing to keep in mind was also the need to find a framework that could work cross-platform with MacOS, Windows, and possibly Linux (as it was not nearly as common). Electron could handle all this, and builds for MacOS, Windows, and Linux.

In theory, a desktop application could work best in this situation, but we want something lightweight and easy to manage. Having a download for a whole application on your PC just for something you may not touch very often along with having to always leave it open in the background for it to notify the user just may turn out to be a little more excessive than necessary. In reality, something like a desktop application may just be another hassle for a user to worry about, which would be a downside when a customer would consider this as a supplement to their daily life.

7.4.4 Mobile Application

A mobile application was another great option for handling data and allowing easy changes to a system for a user. It was something that was with people at most times and could also allow for easy notifications without having to rely on sending out a floor of texts a day to remind someone. Instead, you could simply send out a notification reminder from the app.

There was also a ton of support and options on the best way to create and organize a mobile application. There was React Native, Flutter, Swift, vanilla Java, and Xamarin, along with a slew of other options. We also are given the option to build a mobile app for iOS or Android, or both. This could be used to guide the decision for what framework or language to use to best fit the design of Pill^3.

Something to consider was the need for internet access from your device. However, in modern times, most people have unlimited WiFi and internet from their cellular device, so it does not have to be considered a negative to this.

Something to consider was the use of cross-compatibility, which would be discussed more in the next section. When working with mobile and desktop applications, you need to make sure it could be built for the most popular platforms out right now.

7.4.5 Summary

Mobile Application would allow the most flexibility due to people having it with them everywhere, it allows for simple notification systems. Compared to the other platforms, mobile was something everyone would have on hand and it was the easiest for any consumer to just download an app.

Although it requires a backend, it makes up for the ease of use for any consumer along with ease of implementation and expansion if anything else were to be made. Native screens Are not very modular compared to the other choices which make it much more difficult to work on. Although a Web Application would work really well due to its support, modularity, and ease for a consumer to use, it doesn’t allow for easy notification. Thus, it seems the best option was a mobile platform.

Table 11: Platform

Platform	**Requires Internet**	**Requires Backend**	**Notifications**	**Support**	**Modularity (Ease of expansion)**
Native Screen	No	No	Around device	2/5	Not very modular
Web Application	Yes	Yes	Around PC	5/5	Very modular
Desktop Application	Yes	Yes	Around PC	3/5	Somewhat modular
Mobile Application	Yes	Yes	Everywhere	4/5	Very modular

7.5 Framework/Programming Language

So now that the platform was decided, now we could find a frontend framework/library to use. A framework and library provide a way of programming software using application-specific software. It also provides ways to build and deploy applications.

A framework helps in saving time, special tools, a lot of repetitive code, and more advanced actions done much quicker than if everything was in its own native language.

We need to look at the modularity of the framework, as the packages we could find could heavily influence the difficulty of the project. It would also help simplify the process of making the application. We need to look at what OS the frameworks build and deploy to, how easy it was to program in, and also just general opinion. A lot of the mobile (as well as web and desktop) frameworks also come down to preference.

7.5.1 Java/Kotlin

Java/Kotlin was used for Android development natively. It was pretty low-level and used heavily in Android-only applications. A few years ago, Java switched to Kotlin which allowed a lot of Android apps to be more easily ported over from Android to iOS, and even web development. Kotlin was a very popular language for Android developers and used widely in the industry. Kotlin also was good for security. It also uses native language to program applications because you have more control over how the app functions with the device.

Another thing with Java/Kotlin was that it demands very strong OOP concepts. Using Objects to build components, then using the components to build the framework of the app. There was also no need to access more native APIs for Android.

In most cases, Java or Kotlin should be used if your goal was to build in-depth internal API calls, heavy services, or other intense system usage. However, in many other cases there are few reasons to use Kotlin.

The big thing here was that using Kotlin/Java only works for Android. This can’t be automatically ported over to iOS which was a big thing that this project was looking for. It’s also more difficult as no one else on the team has used Java/Kotlin for programming an Android device which in and of itself was already a very difficult task to pick up.

7.5.2 Swift

Swift was easy to learn and one of the best apps to use to program iOS devices. It has an amazing interface and very powerful native functions to use for iOS. It also has a custom IDE (Xcode) which was meant to work seamlessly with Swift.

Swift has many safeguards made to deter all unsafe code and syntax was clean and simple. Swift, just like Kotlin/Java, was object-oriented so it was easy to manage the use of components and build the framework of the app.

There are some downsides to the language such as lack of support. Swift has been around only about 6 years, so there isn’t much on it compared to Java, Javascript, or Objective-C. It’s also open-source so it’s a smaller community taking care of the project. Apple heads Swift, but there’s a team that manages it.

One of the biggest issues (just as it was with Java/Kotlin) was that Swift was made for iOS only. It could greatly scale to any MacOS such as iOS, MacOS, tvOS, and etc. However, for this project the most important thing was that we could scale it to both Android and Apple devices for easy testing and reaching the most consumers.

7.5.3 React Native

React Native was a multi-platform Javascript development framework that was used for building Android and iOS apps both under one codebase. It was made by Facebook and was still maintained to this day. React Native uses a Javascript syntax extension called JSX which allows HTML tags inside of Javascript which make making components really simple.

The good thing about React Native was you are able to combine multiple sets of logic into one. Opposite of HTML, CSS, and Javascript, JSX allows users to combine all of the logic from all 3 sets of web design into one. This makes all the code very legible, easy to read, and little need for others to reference a bunch of different pages to see how someone has chosen to lay out a page.

React Native also has hot-reloading which makes testing anything on the app much easier. With hot-reloads you could easily make changes and see them real-time without having to restart the entire application to see a change of any size.

The performance was also amazing as it uses threads as close to natively as possible on both iOS and Android. This does not deteriorate any performance compared to using Swift or Java/Kotlin.

However, React Native cannot use internal APIs like Swift or Java/Kotlin due to it being multiplatform.

Now there are also a few downsides to React Native, the main one being that it does not work great with resource-intensive applications, which was much better suited for native programming. React Native was also relatively new to the programming world so in the long-run, it’s a big commitment with Facebook’s team to manage and maintain React Native so that the code does not become obsolete.

7.5.4 Flutter

Flutter was a multiplatform Dart development framework that was used for building Android and iOS apps both under one codebase. It was made by Google and was still maintained to this day. Flutter was extremely new at about 3 years old, so support was not as great as React Native or the Native programming languages.

Flutter uses Google’s own UI library; thus, the components don’t look like their native counterparts, but rather, another Flutter app with Flutter components. They use Flutter components for both iOS and Android, but there are parts of each that are made to mimic native components such as Google’s Material for Android and Cupertino for iOS. With the updates to iOS and Android, these UI designs would remain the same and any design changes require the developer’s input to change and not with just native updates.

With Flutter we also have hot-reloading, native performance, and would not slow down because of the framework. Just as React Native, Flutter was very good at handling native speeds due to how it mocks native threads, but also not the best at handling resource intensive stuff such as using the OS for anything CPU-intensive.

Just like anything else, Flutter does have its downsides. Its OS interactions are below standard, and you don’t really want to use Flutter if you are doing any heavy computation or using a ton of hardware on the phone such as the speakers, camera, or sensors. Google also controls Flutter, so they have complete control over Flutter’s state. This was unlikely, but still something to keep in mind. Flutter’s apps are also generally larger than native and React Native’s applications.

7.5.5 Summary

React Native was the choice to go with. It has everything that was needed to create a great looking app on the frontend, and easily connect to a backend that works well. The hot-reloading, performance aspects, smaller app size, along with just general React Native simplicity in it’s usage makes it the best choice for this project.

Another huge thing to consider with this project was that a lot of time front end frameworks/libraries are based on pure preference. Some of us in the group have used Javascript for years along with React Native and are very comfortable with it while none of us know Dart or Flutter.

Immediately we could avoid native programming like Java/Kotlin and Swift (we also don't have a MacOS device to be able to make a Swift application), with the difficulty behind making native apps compared to multiplatform applications there was an obvious choice of Flutter or React Native.

Table 12: Frontend Framework/ Language

Frontend Framework/ Language	**Android**	**iOS**	**Hot-reloading**	**Ease of Use**	**Support**
Java/Kotlin	Yes	No	No	3/5	3/5
Swift	No	Yes	No	4/5	3/5
React Native	Yes	Yes	Yes	5/5	5/5
Flutter	Yes	Yes	Yes	5/5	3/5

7.6 Database

7.6.1 Google Firebase

Firebase was a mobile and web application development platform by Google that has several integrated tools for analysis, development, and maintenance of mobile applications.

Firebase boasts a lot of advantages, some being the documentation which was bountiful and all online as Firebase has been around for 10 years since 2011. Once Google acquired it in 2014 and more people started using it, documentation just became more and more common for firebase. Firebase also has great performance, great integration, and wide functionality behind it. It was great for both large applications and large applications. In the case of this project, storing data that won't be extremely complicated was well done in a NoSQL database as it's just a bunch of data relating to each device being used. This was a part of the Firestore Database which stores collections that could hold various types of data.

There also exists Authentication (Firebase Auth) which allows users to log in and provide an experience special to them. This helps in keeping track of logins, and keeping users logged in even after they leave the application.

The free plan that Firebase offers was plenty for creating a prototype like we are for this project. Although there aren't as many tools as there are in AWS, Firebase holds all of what was needed for this project.

One of the best things about Firebase was the dashboard, which helps manage all of the hosting the application, storing data, and securing login information. The dashboard was extremely simple to use and offers a full GUI interface which makes management and analysis of how the backend was handled really simple and easy for the developers to do.

7.6.2 AWS

Amazon Web Services was pretty much the world leader in cloud ecosystems and cloud management made by Amazon. AWS has tons of features, all really well done. There are several tools for anything you would need done from Quantum calculations to improved IoT Machine Learning and Artificial Intelligence to hardware devices. There are so many services that sometimes AWS could get a little oversaturated for a project that leans on the simpler side such as this project.

AWS's database equivalent to Firestore Database (NoSQL) was AWS DynamoDB. The free tier includes a ton of storage and requests per month so it would be a great option to include on this project as it has so many uses. Another option was AWS Aurora with a MySQL and PostgreSQL-compatible relational database.

In this case, because information was being stored in relation to a user which could be a large slew of data, we would prefer a NoSQL database so we could better store data in regards to each user.

AWS's dashboard was useful and gives the user a lot of information on how everything was running, but there was a lot happening. The learning curve was pretty steep and again, for something as simple as this project, AWS may be excessive to include on this project.

Naturally, AWS was best suited for larger applications that could utilize AWS's features to the max. With a smaller project that isn't scaled (or planned on being scaled as of now) there was no point to use a bigger-scale cloud ecosystem that would overcomplicate a lot of the project's technicalities when something simpler could be used to make the developer's lives easier without affecting the consumer's experience.

7.6.3 Microsoft Azure

Microsoft Azure was Microsoft's answer to a cloud ecosystem; just like Amazon Web Services it has a huge range of options when it comes to what services it could provide a project or company.

Microsoft has its own NoSQL Database called NoSQL. NoSQL could be Key-Value, Document, Columnar, and Graph. In this project's case we are leaning more towards the possibility of using Document NoSQL as it was a simple way to store and fetch lots of information in regards to a simple user. Each "Collection" would contain a bunch of relations to other data known as "Document" in Microsoft's NoSQL Database. They also have a SQL Database called SQL Database; however, also previously stated our project would most likely call for a NoSQL database.

The dashboard also is well done and allows for easy management of how the entire infrastructure is doing. A popular consensus though was that Azure's dashboard and/or interface feels very overcomplicated, which was a big thing this project was trying to dodge. Our project mainly revolves around the hardware and the software (mobile application, backend, and database) should all be quite simple and easy for the developers to use and reference.

The free plan of Microsoft Azure contains decent usage, roughly the same as both of its competitors. As this whole backend and database was a supplement, there should be no need for us to ever surpass the free tier of Azure's limits.

Azure's whole system falls into a similar trap as AWS's services on a smaller scale. Microsoft offers so many different services that the whole system could become too complicated for a simple project to use. It seems much better suited for bigger projects that would scale up and take advantage of its infrastructure, IoT, and other cloud data storage options.

7.6.4 Heroku

Heroku was a Platform as a service (PaaS) that was managed by Salesforce. It was a lighter weight alternative to all of the major cloud providers by scaling down the amount of services it offers. People commonly host their smaller-scale applications on Heroku as it is a cheaper alternative for small scale applications that require a platform.

They offer both a NoSQL and SQL database which are NoSQL and PostgreSQL respectively. As stated, the SQL database doesn't pertain to us. The NoSQL database Heroku supports covers Key-Value, Document, Columnar, and Graph. Document would be most useful to us to store user data easily and efficiently.

Heroku's dashboard was almost nonexistent. In theory it works perfectly fine, but in comparison to the other options that are presented Heroku has a very simple and minimal dashboard. Users could see their connection, and see "Dynos" which are Heroku containers to run and scale apps. Placing the apps in these containers allows Heroku to utilize virtualized Linux containers and help make everything under the hood with Heroku run quicker.

Heroku's free tier was subjectively very bad. At minimum you would be required to spend their monthly fee to have hosting. For a prototype, there are plenty of ways to dodge having an extra cost; other services have more features and still could stay cheaper than a smaller-scale hosting platform.

Heroku was one of those hosting platforms that are perfect for companies that need a beautiful in-between. Places like Amazon Web Services, Azure, and Google Cloud might upcharge past free tier, Heroku was the right in-between when someone wants quality hosting, but doesn't need the extensive and expensive features of major cloud ecosystems.

7.6.5 Summary

Google Firebase was most likely the most reliable option. Firebase has plenty of features that are super helpful in regards to this project; Firestore and Authenticator are two of the features that would be extremely helpful.

The free tier of Google Firebase was still good quality, plus if for some reason the application ever scales up, Firebase has that possibility of being able to scale up while Heroku may not do that.

In regards to complexity, Microsoft Azure and Amazon Web Services both are too complicated for a supplement to a project such as Pill^3 using a database to store the times that people are going to be taking pills. Authenticator could be used for each person's login- it would make it easy to keep track of who's machine it was connected to because Authenticator has that feature built-in automatically.

Table 13: Summary

Data	**Offers NoSQL**	**Free tier**	**Infrastructure Complication**
Google Firebase	Yes	Yes	2/5
Amazon Web Services	Yes	Yes	5/5
Microsoft Azure	Yes	Yes	4/5
Heroku	Yes	Yes; but at a much lower scale	2/5

7.7 Backend

7.7.1 Node.js

Node.js would be used as the backend as it was just the most common and one of the largest runtime environments to be used today.

Node.js aids in both backend and frontend development in coordinating both work environments. For this project, Node.js would be used mainly for backend development, where it would make whatever calls it needs to store and fetch data from the database when a user needs to access and modify it when they take their pills.

Node.js handles a lot of different size projects, but excels in smaller projects where less was needed to be fetched from the server. In our case, we won't be fetching a lot from the server so Node.js was a great solution to this problem.

7.8 Software Design

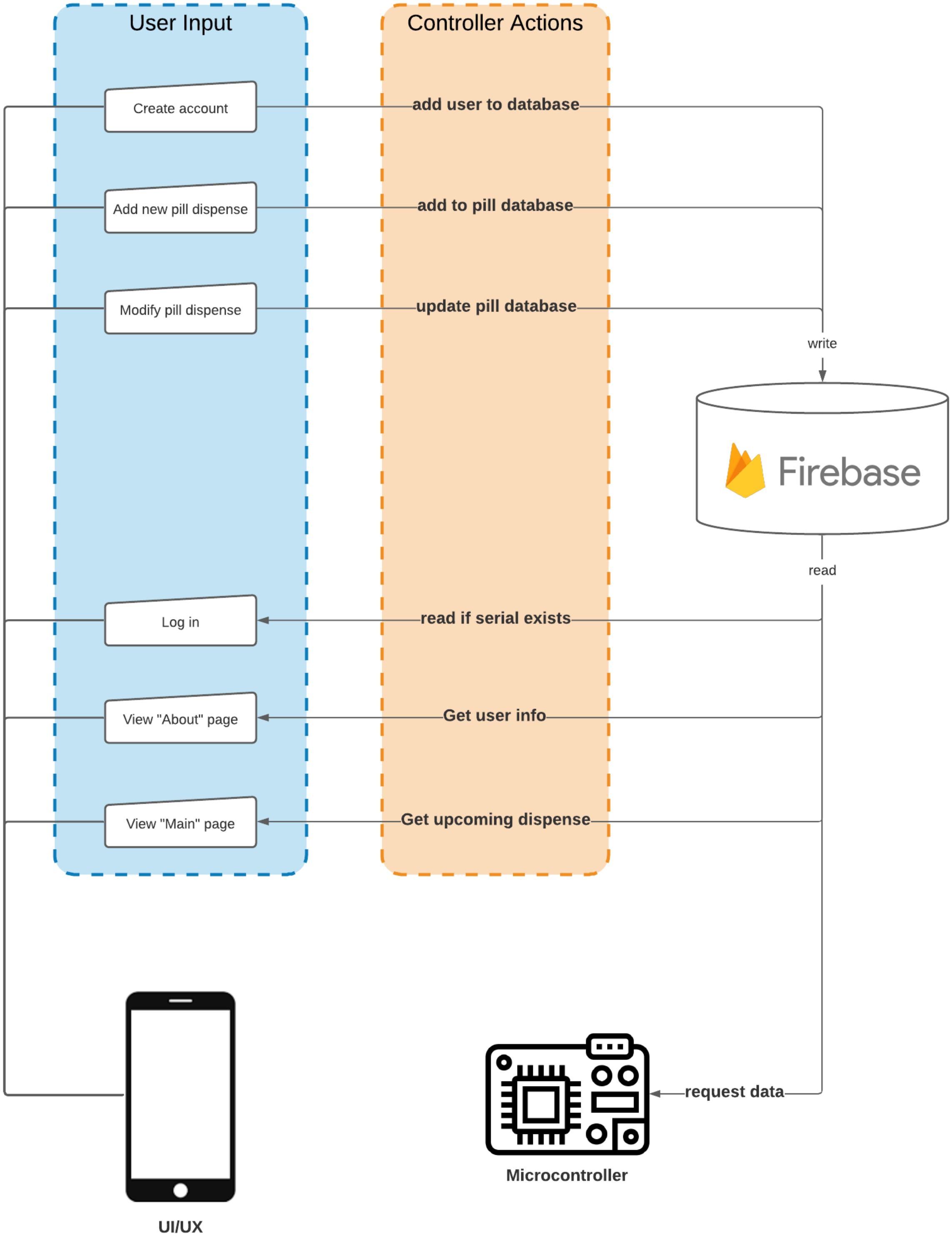


Figure 10: Software Block Diagram

UI/UX Prototype

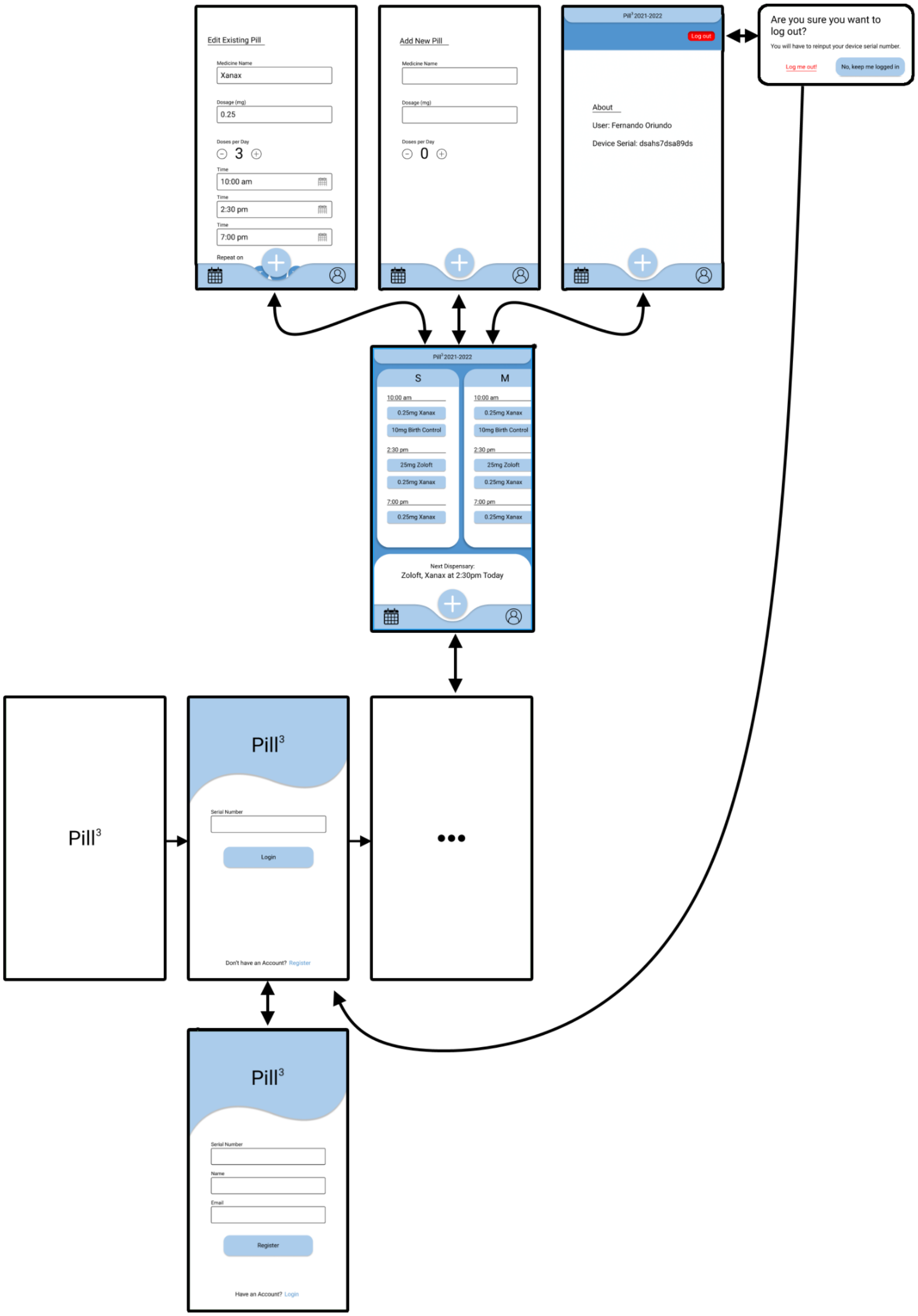


Figure 11: UI/UX

7.8.1 Frontend Design

UI/UX Design

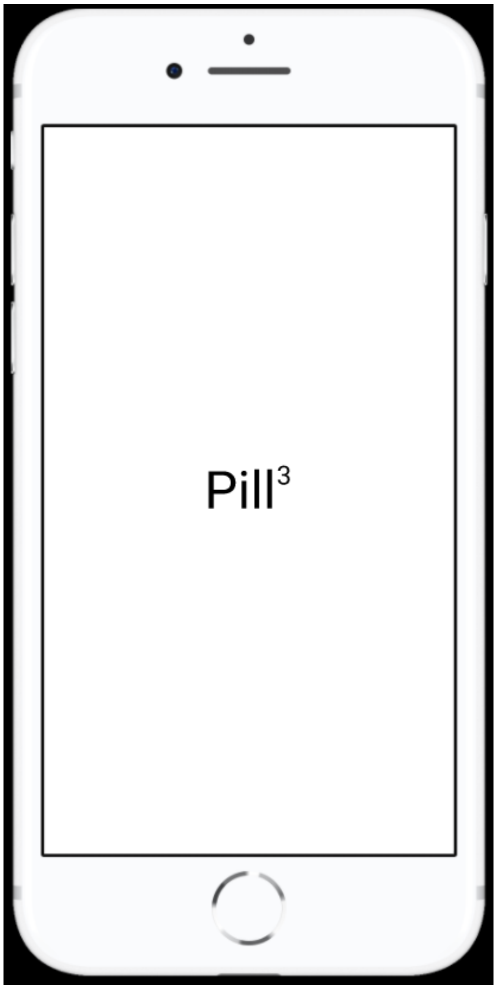


Figure 12 : Splash Screen

The splash screen was probably one of the most important screens. It was the first thing someone sees when opening the app to leave a good impression. In this case, we opted for a simple splash screen that presents the user with our logo while the next page loads. This was another way of saying “welcome” to the user and showing them that their application was loading. If it was too busy, the user could get anxious and feel as if too much was going on.

Another important aspect of the splash screen was for branding. We want to push our logo so that our users remember it as much as possible. By putting our logo as the first thing users see, it was very clear that we care about our branding.

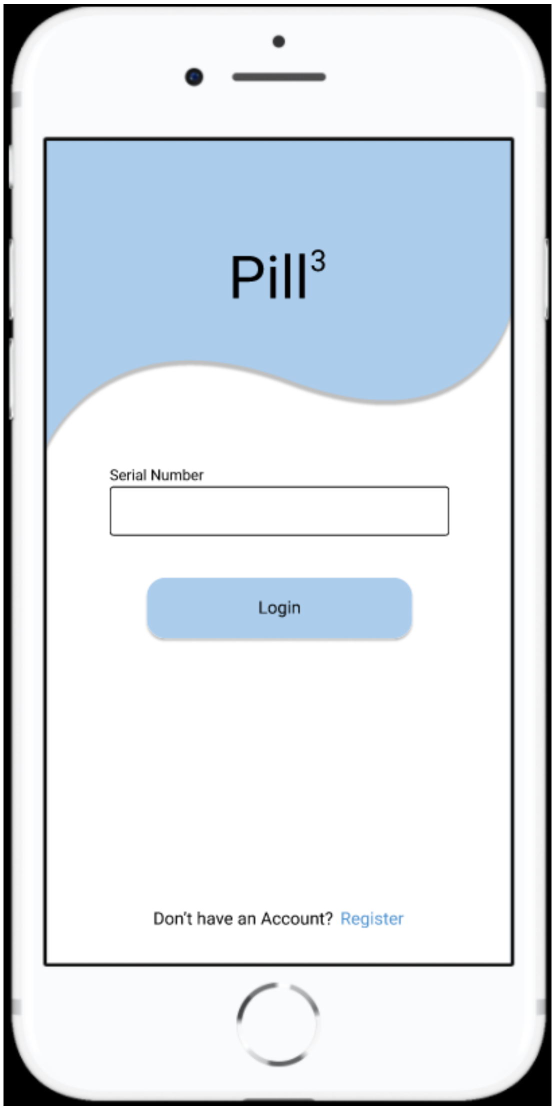


Figure 13 : Log in

This screen would present the user with the logo on top to further push branding as well as asking for input for their device's serial number. Using a serial number ensures users don't have to rely on a username, email and password. It also makes it easier to navigate the device and easily manage log-in.

This serial number log-in isn't really a security burden as the serial number would be a string of alphanumeric text that would be too difficult for someone to guess. This also voids the chance of anything getting leaked, as anything that would be leaked was low-level security risk and would just be the serial number of a random device.

You could also go to registration in the case that you are new to the application.

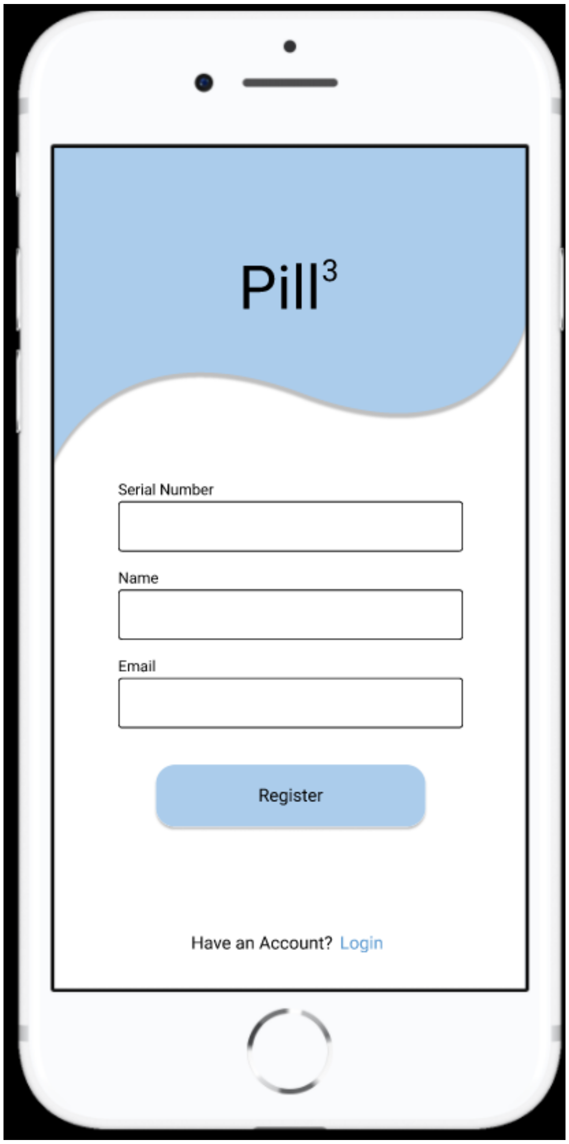
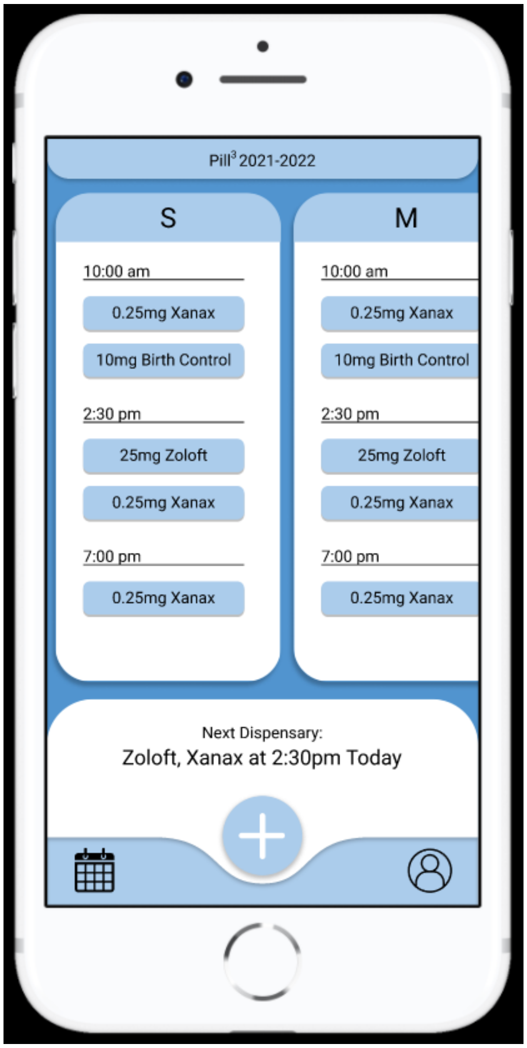


Figure 14 : Registration

The registration page allows a user to create a new account with Pill3. It requires your device's serial number, name, and email. This allows a user to use email to retrieve serial numbers if lost, have their names stored in the system for management purposes, and registers your device's serial number to be associated with your account.

In the case you already have an account, there was a link at the bottom to go back to the login page.



Pill^3 2021-2022

S	M
10:00 am	10:00 am
0.25mg Xanax	0.25mg Xanax
10mg Birth Control	10mg Birth Control
2:30 pm	2:30 pm
25mg Zoloft	25mg Zoloft
0.25mg Xanax	0.25mg Xanax
7:00 pm	7:00 pm
0.25mg Xanax	0.25mg Xanax

Next Dispensary:
Zoloft, Xanax at 2:30pm Today

Figure 15 : Main Page

This screen was the main screen where the user could see their weekly pill schedule, their next dispensation and when that would happen, and the navigation to other parts of the application.

One of the most important features introduced in this page was the viewing weekly schedule and showing next dispense. The application's main purpose was to notify the user when it was time to take the pills and also easily keep track of when that is. In this case, it was clearly laid out to the user in an easy-to-follow format.

There was also the navigation menu at the bottom of the screen, this was where the user could add a new pill to their schedule or navigate to the account screen. Take a look at the components section of the document for more information on how it would work.

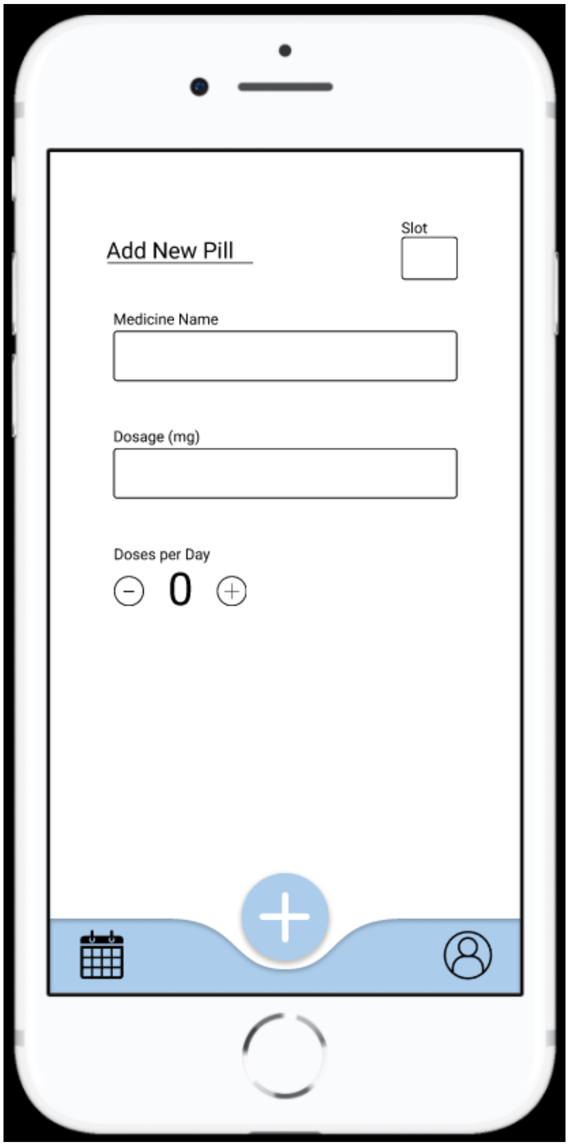
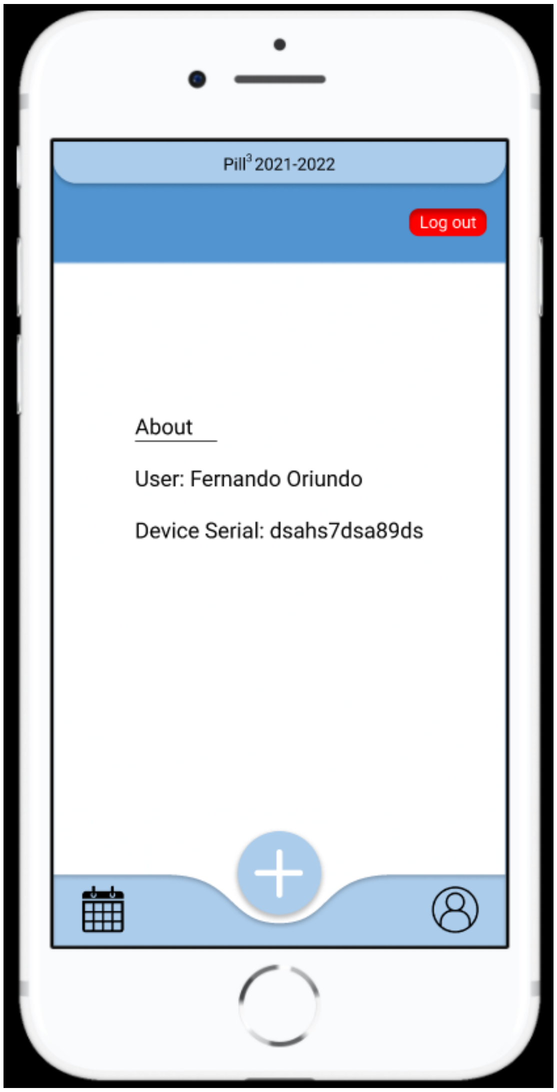


Figure 16 : Add New Pill

This was where users could add new pills to their schedule. They would use the same textboxes as the log-in screen (see the components section for more on textboxes) to ensure the app flow remains constant throughout the entire app and to ensure structure.

The doses per day are also listed here, which was dealt with in more detail in the components section. Here, users could choose a time and what days they want these pills to dispense and could scale up to 4 times a day. The "Add New Pill" button only shows up when the dosage was set to more than 0 a day.



Pill^3 2021-2022

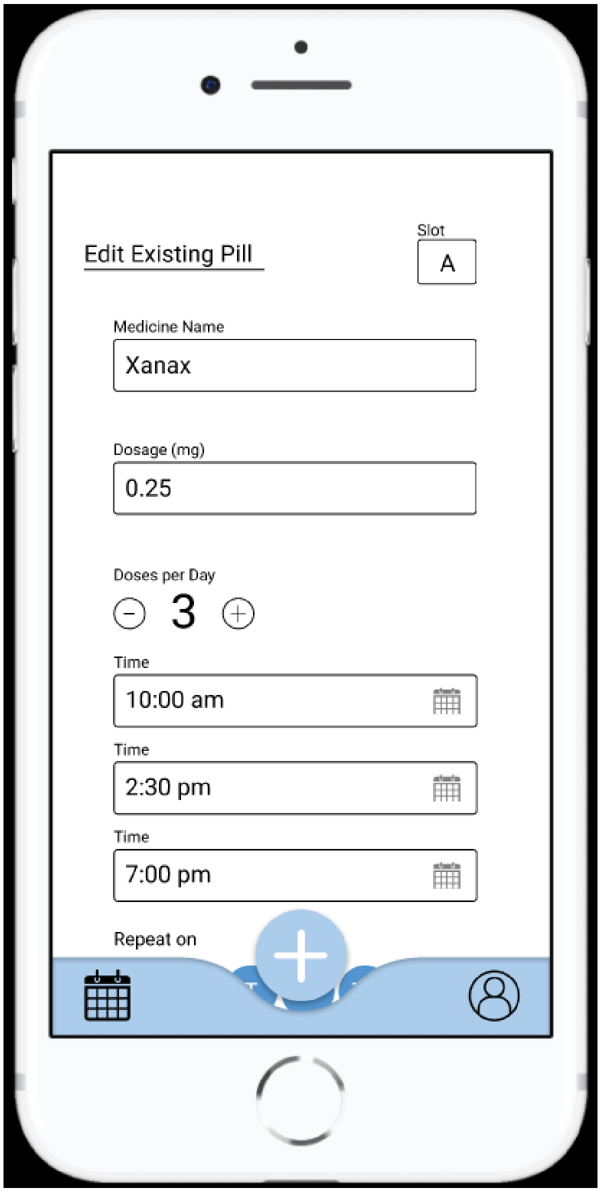
About

User: Fernando Oriundo

Device Serial: dsahs7dsa89ds

Figure 17 : About Page

This page shows the user the person, device serial, and allows the user to log out. There was not much important information here as there isn't much data for the user to be able to view.



Edit Existing Pill

Slot: A

Medicine Name: Xanax

Dosage (mg): 0.25

Doses per Day: 3

Time: 10:00 am

Time: 2:30 pm

Time: 7:00 pm

Repeat on

Figure 18 : Edit Pill

This page was very similar to the “Add New Pill” page, except it was filled out to the selected pill for a user to edit. To cancel this operation, all you do is click back to the calendar on the bottom left.

Just as before, if dosage per day was set to 0 the user would not be able to press the “Edit Pill” button.

React Native Components

React and React Native both work on the idea of components. These components then build up to create the application. There are design choices in both user interface and user experience that were made to maximize the experience of the application for the user base.

In this section, we would go over some components that would be used to build the application. This includes React Native libraries, frameworks, and other frontend tools used to develop an efficient frontend.

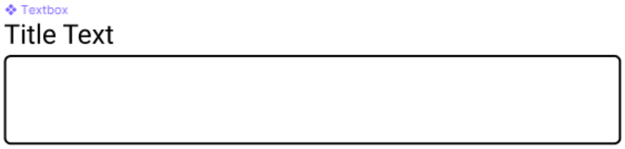


Figure 19
Textbox to provide an input to a form for submission.

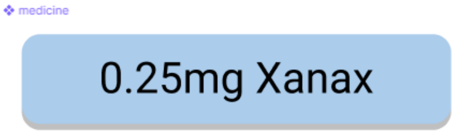


Figure 20
The pill boxes that would show on the main page. This provides the user with a visual representation for when the pills would be dispensed. Users could also press this button to go into the pill edit menu where they could modify when these pills are taken.

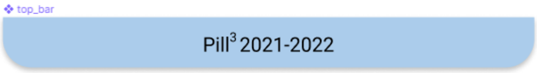


Figure 21
This was just the top bar that would represent branding for the device. This would show up on all the pages that the navigation bar shows up on.

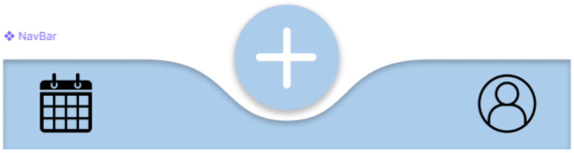


Figure 22
Users could use the navigation menu to go to account details, add a new pill, or go to the main page. This should display on every page on the device for simple navigation.

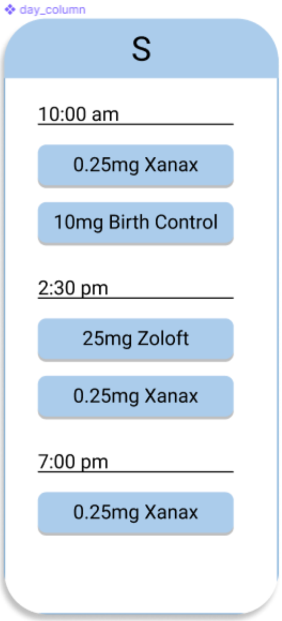


Figure 23
This column was how users would see the pill boxes and have a visual representation of when the pills would be dispensed.

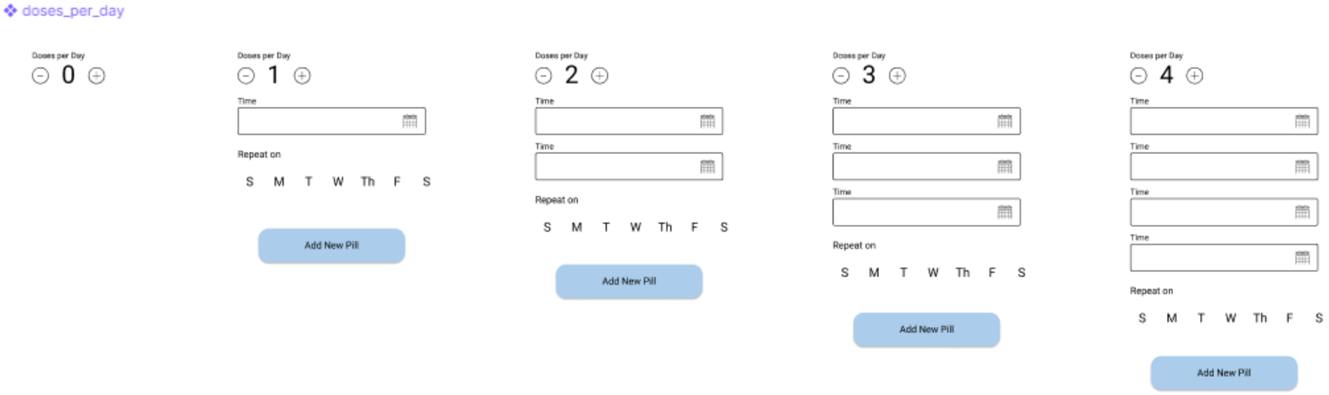


Figure 24
This collection of components are for adding or modifying a pill and when it was taken. The list should be dynamic so a user could input the pill name, dosage, how many times a day the pill

was dispensed, and what days to repeat the dispense schedule. There should also be an input to tell the device what slots this pill was in.

7.8.2 Database Design

Database Structure

Firebase by Google was the technology being used for data storage in Pill^3. You could read more about why Firebase was chosen, and the features it has, in the Software Technologies section of the document.

Firebase uses a NoSQL database storage method which stores data as JSON objects. This could be leveraged to efficiently store user data and easily store and fetch the information necessary for the application.

Specifically in regards to data storage in Firebase, we would be using Firestore, which allows for this simple storage method of essentially cloud-stored JSON files. Firebase also offers a real-time database which stores data in a single JSON-like file. This would also work for this use-case; however, Firestore offers collections which simplify the process of navigating users and makes data fetching and organization much quicker. Using Firestore allows for structured documents compared to Real Time Database's simple JSON-tree. This would become more clear in the examples provided below.

While thinking about database design, we first need to acknowledge elements we need to have stored:

- Device's serial number
- User's email
- User's name
- Pill dispensing
    - What pill
    - What slot to dispense from
    - Dosage
    - When to dispense
        - What days to dispense
        - What times to dispense on each day

Firestore allows for collections, documents, and fields.

Collections contain documents.

Documents contain either collections (subsets) or fields.

Fields are composed of some data type:

- string
- number
- boolean

- map
- array
- null
- timestamp
- geopoint

String, number, boolean, and array would most likely be what was used. Although there was a “timestamp” type, it was best for length of time for easy conversions to other lengths of time, marking the current time, or more accurate and precise ways of representing times- this was something we don’t need to consider as it would only be used to keep track of what time the user sets without a need for conversions.

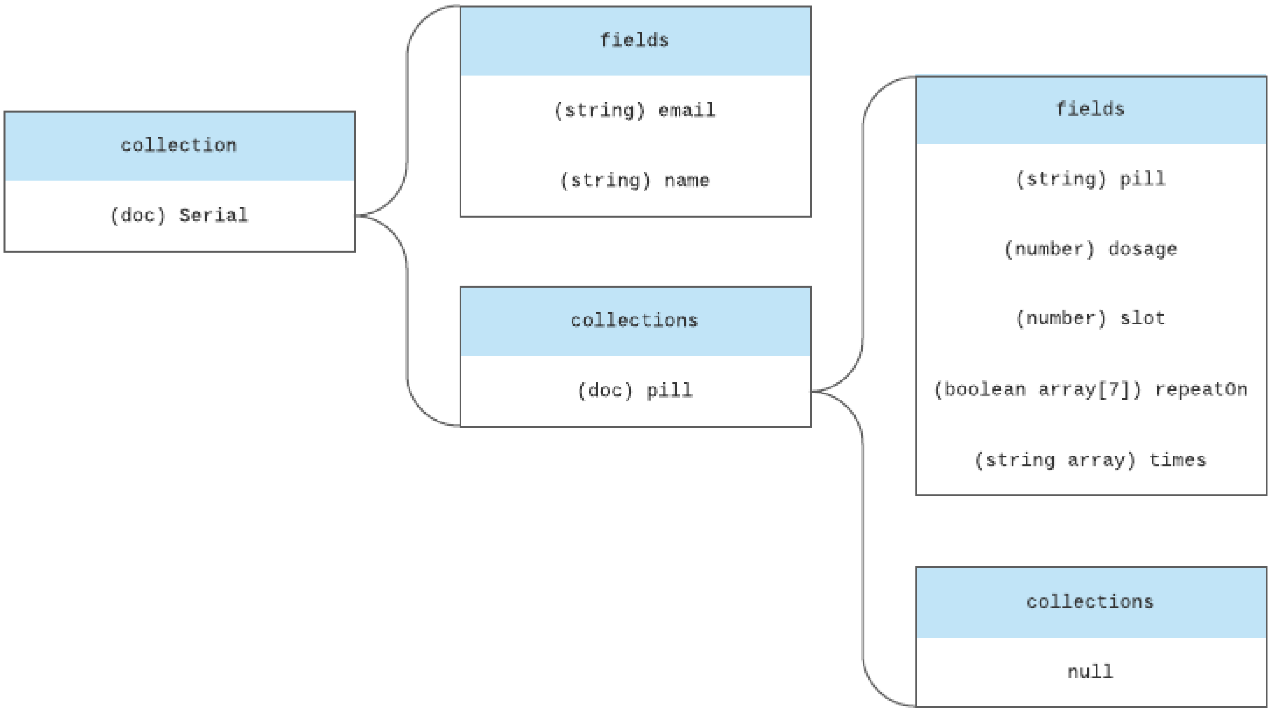


Figure 25: Fields

For the fields for each pill there would be these fields:

- pill: the name of the pill
- dosage: how many mg was in the pill
- slot: what slot this pill was contained in
- repeatOn: a boolean array of size 7 (each day of the week)

[0]	[1]	[2]	[3]	[4]	[5]	[6]
Sun	Mon	Tue	Wed	Thu	Fri	Sat

true	true	true	true	true	true	true

Ex. Given only weekdays:

false	true	true	true	true	true	false

- times: a dynamic string array with the times (in 24h clock) to dispense the pills in order from 0:00 to 23:99. This would be in sorted order for easier processing. The app allows up to 5 times a day, so the max array size would be 5.

[0]	[1]	[2]	[3]	[4]
time1	time2	time3	time4	time5
10:30	13:30	20:00	null	null

The reason for this design was to easily section each device by serial number, as the serial number would be unique. Although a popular way of organizing data was by email, it was fully possible for someone to have many devices tied in with their email.

Cases include:

- A caregiver that wants to manage many different devices for clients
- Someone gets a new device and cannot reuse the old one, requiring a new account
- Multiple people in the family who use the device, but cannot access technology

7.9 Summary of Design

7.9.1 Exterior

The subsystems and software described above would be used in conjunction with each other to create a working pill dispenser that could monitor the amount of medicine left inside, inform the consumer of any status changes or updates, and precisely deliver the medicine to the user at user-chosen times. To achieve all this, the subsystems need to be arranged and designed in a way that makes them both intuitive for the user to interact with and reliable in regards to storage conditions and timing. The entire design would be housed in a cubical wooden frame with rounded edges. Of the six faces on the cube three would be featureless panels, (when observing the device from the front) the left and right panels and the bottom panel. The three remaining panels would be featured panels. The front panel would be the UI panel, the back panel would have the power IO port and air ventilation grid, and the top panel would be the medicine storage access panel.

The front facing panel of the cube would have a rectangular opening along the bottom, with a small stand emerging from it. A cup would be placed on this stand, and would be the location that the pills are distributed to. To achieve this, the stand would have a weight sensor in it, and a funnel would be situated above it. The sensor would notify the system when the cup was in position, and the medicine would be dispensed through the funnel and into the cup for the user to take. The front panel would also house the display used to notify the user about upcoming dosages, medicine supply levels, and any other pertinent system information. The final feature of the front UI panel was the speaker which would alert the user through audio notifications to any medicine dispensing or system issues.

The top panel would be flush with the rest of the frame except for the medicine storage container access ports. These access ports would allow the user to restock the device with more medicine directly from the bottle. Each port would have a screw-on lid with a child safety lock on it, requiring the user to squeeze both sides of the cap before it was able to turn. The access ports are spaced evenly to look aesthetically pleasing, and to allow for the internal room between storage cylinders.

The back panel would contain a small port to which the power supply would connect the device to the wall, and a grated vent to assist in air flow. The power supply would include an LED light next to the port to indicate whether the device has sufficient power when plugged in. The seal around the power supply would be flush with the wooden frame and allow for the power supply to be disconnected from the device. The vent would consist of a square of slotted lines with a filter and fan covering the inside of the vent. If any other IO ports are deemed necessary later in development, they would be placed alongside the power supply port on this panel.

7.9.2 Interior

The interior of the design consists of the storage cylinders, electrical equipment, and medicine dispensing tubing and systems. The interior support system which holds every subsystem and electrical component in place was made of wood, similar to the exterior of the Pill^3, however this wood was not treated and sanded as it would not usually be visible to the consumer. All interior pieces are held in place by screws and metal brackets. Any wires used to connect electrical components or transmit information throughout the system would be organized and fastened down, with different colors corresponding to different functions.

The storage cylinders would be the subsystem responsible for sorting, storage, and distribution of the user's medicine. Access ports from the top panel would connect to the storage disk for each type of medicine and deposit any pills inserted by the user. These pills would remain in the storage disk until it was time for a scheduled dosage at which point the storage disk would begin to spin, pushing the pills towards the curved sorting edge. This edge would only allow one pill through at a time, and would push the pills along a linear path. After a short duration, the path curves in the same direction that the disk was spinning, which serves to space the pills out appropriately. The pills are then pushed through a progressively larger series of sorting paths that determine the size of the pills, and allow for only one pill of each size to pass in front of the infrared sensor at a timeOnce a pill passes the sorting edge, it would continue to be moved by the spinning disk along a path, until it passes the infrared sensor. Once the infrared sensor was triggered, it would count the pill as distributed and the disk would stop spinning to prevent any additional pills from being dispensed. The dispensed pill would continue along the disk until it falls off the edge and into the distribution funnel, and would be dropped into the awaiting cup.

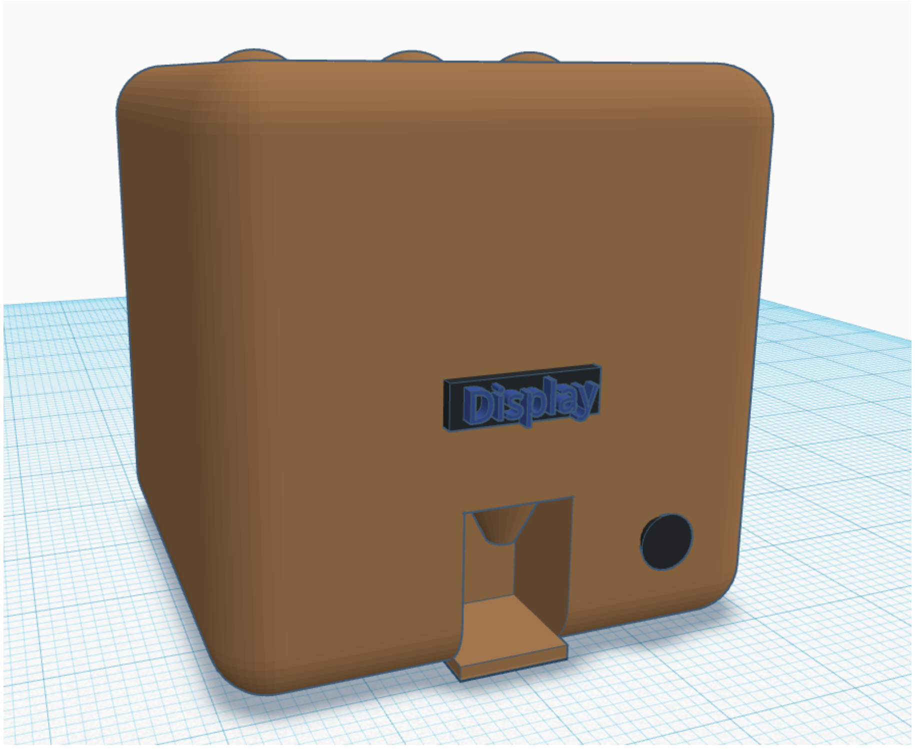


Figure 26: Initial Design for Pill^3 Exterior

8.0 Project Prototype Construction and Coding

For our device, we would be implementing two PCBs, or printed circuit boards, so that our device could function smoothly. The PCBs play a crucial role in our device as it was what we would be using to design our power supply and connect the microcontroller to the LEDs, speaker, motors, and sensors.

In order to achieve this, the software we would be planning on using to design our PCBs would be Eagle Autodesk. Eagle Autodesk offers many features that are very beneficial to us. One of them was having the ability to upload libraries from webench onto Eagle Autodesk. Webench was one of the tools we would be utilizing in our design to make our power supply. Webench was a free online design tool by Texas Instruments that was used by many to design accurate power supplies. Aside from designing our own, there are many schematics available to utilize or gather inspiration from.

8.1 PCB Vendor and Assembly

8.1.1 Schematics

At the early stages of the PCB design we looked at different approaches and components to be placed on the board. There were a total of three designs, however, the second and third one are almost the same with the difference in pin connections. First we will talk about how we power the Pill[3], we would be using an AC/DC, this will provide 12V to the board, MCU and monsters. In the PCB we added a voltage regulator that will step down the voltage for the small components to 5V. For the input voltage of 12V we purchased an AC/DC adapter and DC power jack. The AC/DC adapter would convert the AC power from the wall outlet to DC power and the DC power jack would be mounted onto the PCB to power our device. The DC power jack consists of two pins. For our device, we would only be utilizing two pins. Pin 1 to connect to the positive and Pin 2 to connect to ground.

When designing the PCB we had to look for all the input voltage and current drawn by the devices placed on the board. At the beginning we were looking into placing one voltage regulator per component, however, since most components shared the same rated voltage having multiple regulators won't make a difference in design but in cost. In the table below we will be able to see the requirements of each component, we decided on the AC/DC converter based on the total current drawn.

Table 14: Supply Voltage and Current

Component	Voltage	Current
Microcontroller	12 V	20 mA
IR sensor	5 V	20 mA

Motor	12 V	350 mA
Temp. Sensor	5 V	8.1 uA
Weight Sensor	5 V	2.5 mA
Display	5 V	430 uA
Motor Controller	5 V	1 A

In order to select our voltage regulators, we utilized webench. Webench gave us a wide variety of options to choose from. We input our needed values onto the webench and receive the regulators seen below. The topology we chose for our converters varied but it was primarily a buck. Our main requirement when choosing a design from webench was the cost and efficiency. An efficiency of over 80% was chosen at first for each piece of component. Cost was also kept to a minimum. At the final PCB design we only used one of the designs from Webench.

The design we looked into for the microcontroller was based on the supply voltage and current. The microcontroller requires 12 V and 20 mA in order to operate. This would be supplied by the TPS62175DQCR which would be implemented on our PCB. The design could be seen in the figure below. It has an efficiency of 90% and costs $0.91 to produce. Although it was a good option we decided to not place it in the design.

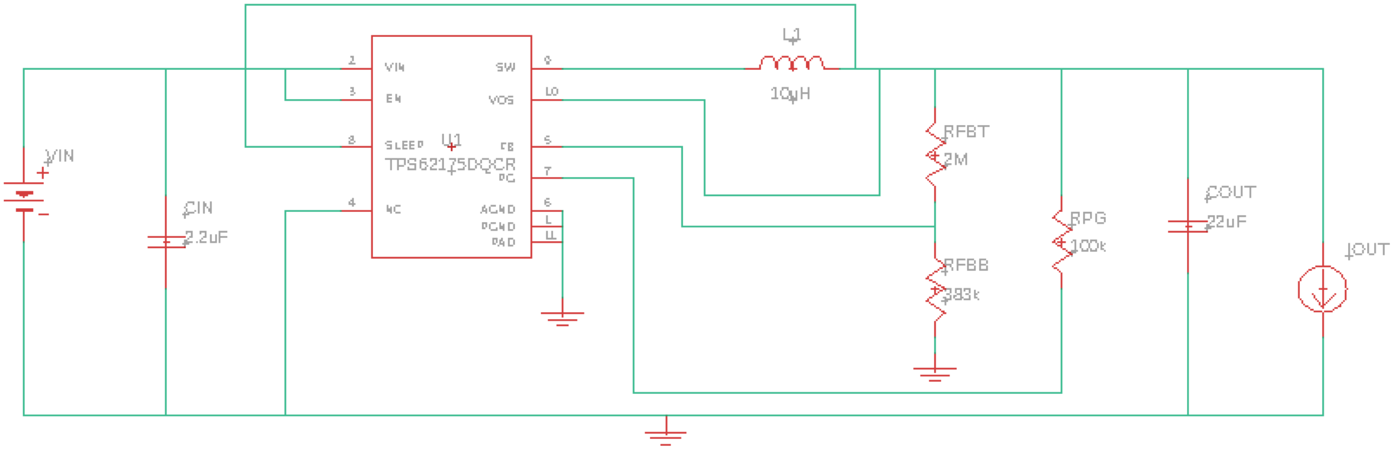


Figure 27: Voltage regulator for MCU

Our display requires 5 V and 430 uA in order to operate. We look into the TPS62125DSGR regulator. This was a 3 V - 17 V, 300mA buck converter with an adjustable threshold and hysteresis. The schematic of the power supply could be seen in the figure below. It has an efficiency of 85% and costs $0.71 to produce. After further research we decided to not include this design due to its complexity and size.

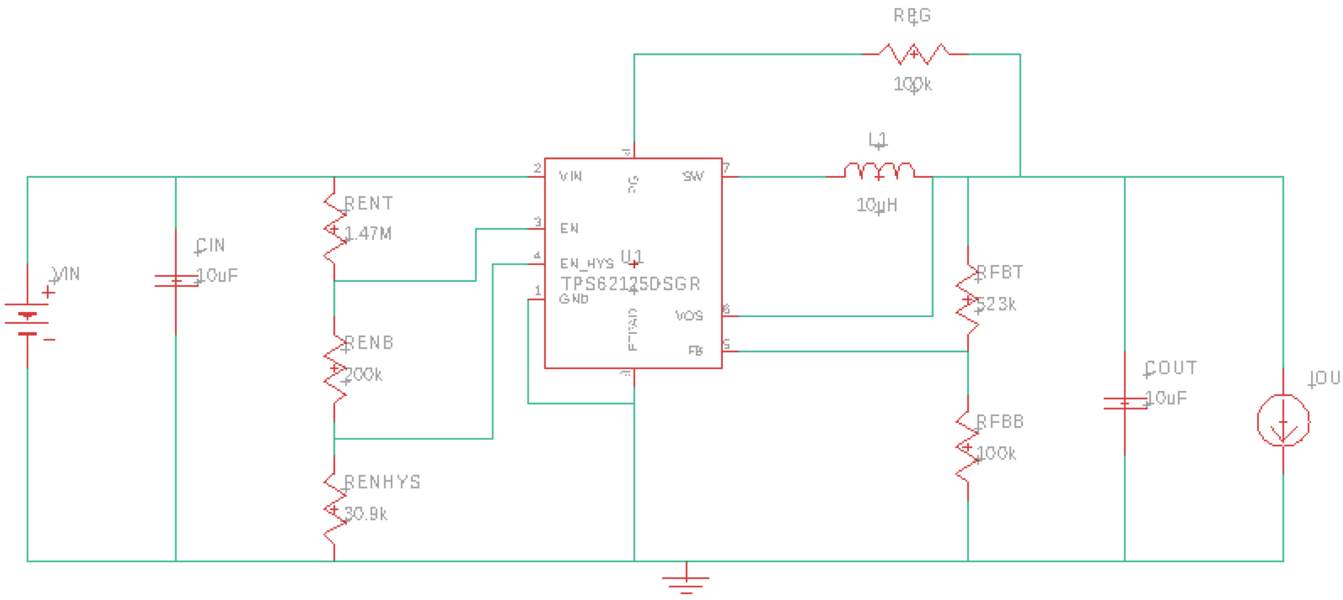


Figure 28: Voltage Regulator for Display

The temperature sensor requires 5 V and 8.1 uA in order to operate. We would be using the LMZ12001TZ - ADJ/NOPB. This was a simple switcher, 4.5 V - 20 V, 1 A DC/DC power module. The schematic for this could be seen in the figure below. It has an efficiency of 89.9 % which meets our standards. Although it was priced at $5.01. After further research we decided to not include this design since it requires too many components.

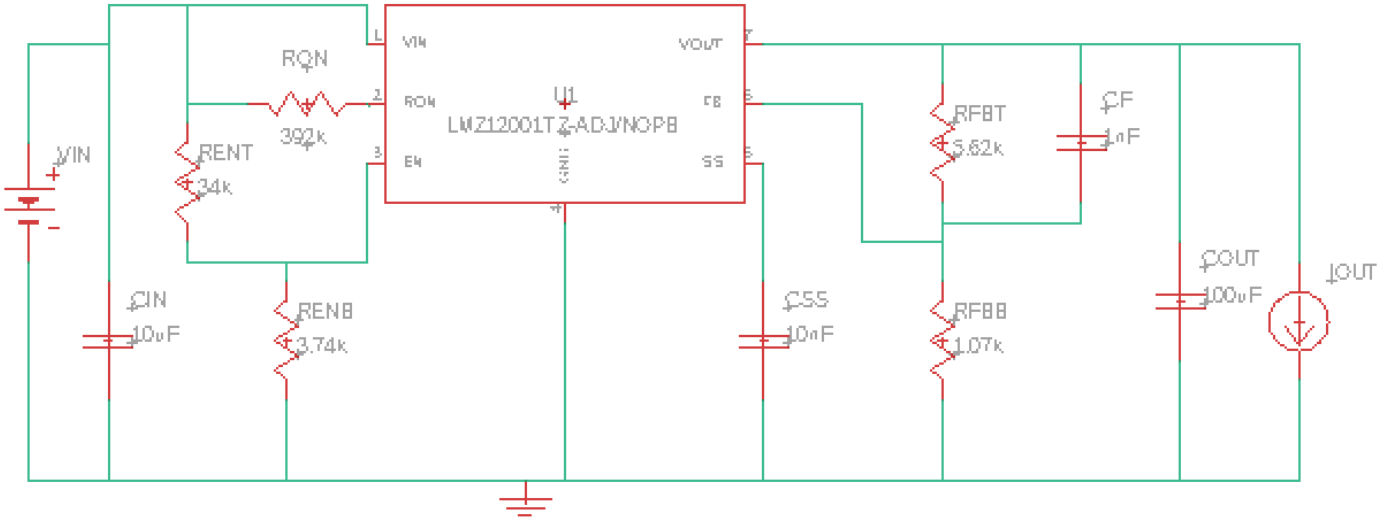


Figure 29: Voltage Regulator for Temperature Sensor

The weight sensor requires 5 V and 2.5 mA in order to operate, although it was a very small device we had to make sure the voltage regulator that we use was able to deliver the appropriate voltage to it. We decided to look into the LM5165XDRCR voltage regulator. This regulator was an ultra low Iq synchronous buck converter with high efficiency over wide input voltage and load current ranges, with integrated high-side and low-side power MOSFETs, it could deliver up to 150-mA current which was delivered to the fixed output voltage . The schematic for this could be seen in the figure below. It has an efficiency of 89.2 % which meets our minimum standards for the Pill^3, although there were designs with a higher efficiency rate, some of them were either higher in price or did not meet our minimum requirements. However, after further consideration we concluded that this regulator was no longer needed.

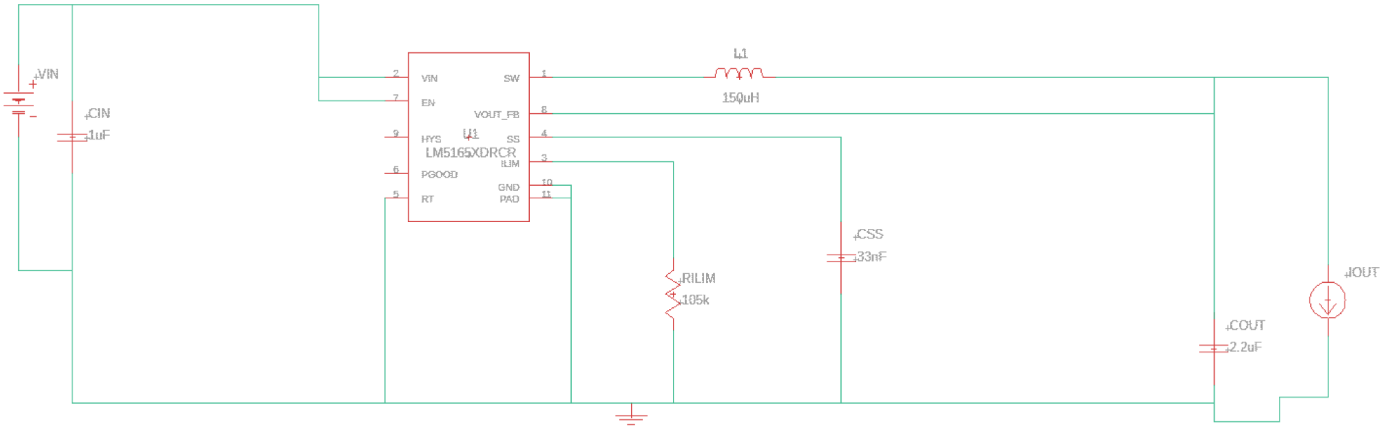


Figure 30: Weight Sensor Voltage Regulator

The infrared sensor requires 5 V and 20 mA in order to operate. We considered using the TPS62125. This was a high efficiency synchronous step-down converter for low and ultralow power applications . The schematic for this could be seen in the figure below. This design has an efficiency of 90.1 % which meets our minimum standards. One of the things we had to consider was that in order to power up our overall PCB we would be using a 12 V DC source since we have specific components that require a high input voltage compared to smaller components, due to this we decided to pick a design that has the flexibility to step down the input voltage for the smaller components . The price for this regulator was $0.70. Other designs were considered for the infrared sensor but this one in particular has the ability to either step up or down the voltage if needed, however, the options provided by the regulator were far too advanced for the overall Pill^3 design.

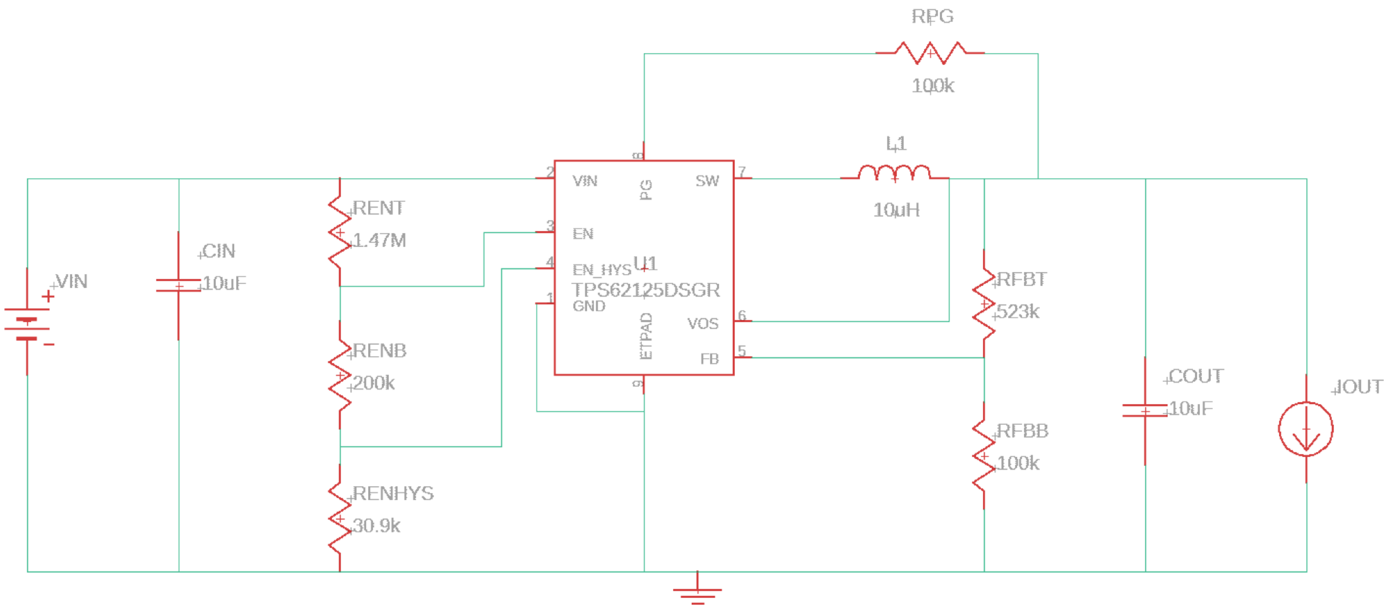


Figure 31: IR Sensor Voltage Regulator

Lastly, the motor requires 12 V and 350 mA in order to operate, when testing for the motor we notice that it could work with as little as 7V, however, the results obtained were not ideal for the Pill^3 due to this we decided to use 12V for each of the motors. Since we would be having three dispensing mechanisms, a lot of power would be needed. We look into the TPS5516x-Q1. This was a high-voltage synchronous buck-boost DC-DC converter. This device is able to provide a stable power supply output from a variety of input-power supply, an example would be something as large as a car battery. The buck-boost overlap control on this design guarantees an automatic transition between the step-up or down mode that provides an optimal efficiency. The schematic for this could be seen in the figure below. It has an efficiency of 93.01 % and a price of $2.98, which stays within the estimated budget of our overall design. Other designs were considered for the motor, since this was one of the key components of the Pill^3 device we had to analyze each of the regulators available to make sure we used the most adequate for the Pill^3 . After further research we concluded that a regulator will be unnecessary for the motors since the motors and even the MCU can be directly connected to the input voltage.

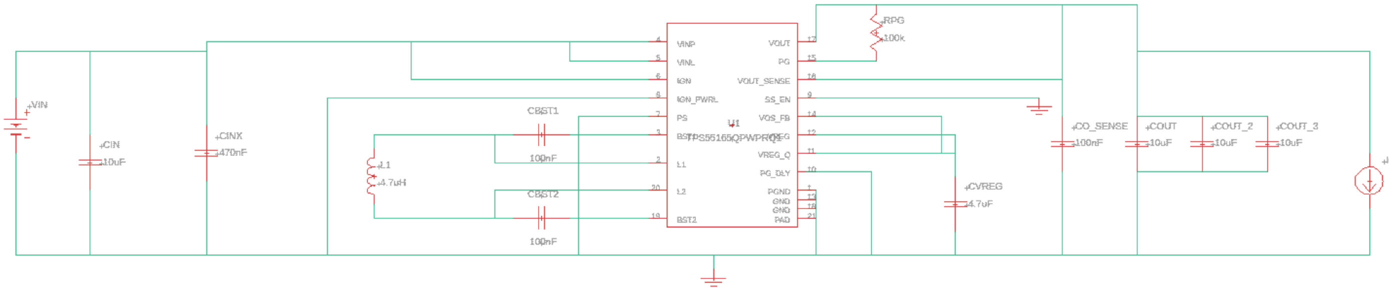


Figure 32: Motor Voltage Regulator

After looking into several options and designs to create a working PCB, we decided to stick to a simple voltage regulator design. This regulator is able to take 12V and supply 5V to all small components in the board. In the figure below we will be able to see the regulator selected for the Pill^3 design and a switch that was added for testing purposes.

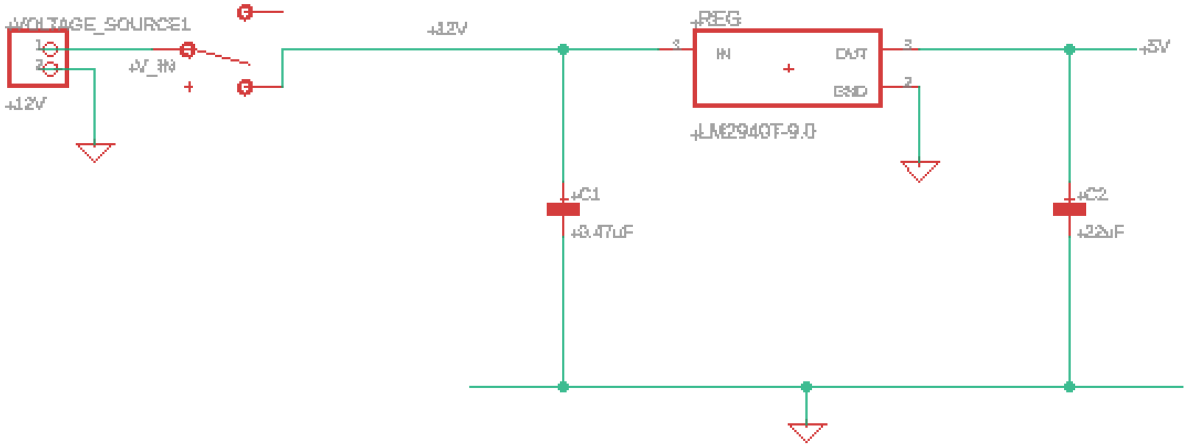


Figure 33: 5V Regulator

8.1.2 Main PCB

In the figure below we can see the multiple designs of the PCB. Here we will see the selected for the voltage regulator together before it's implemented in the main PCB design. In the figures below we will see the connections of the components along the Arduino pins for both the analog and digital pins, we will see the early stages of the design as well as the final implementation used in the Pill^3.

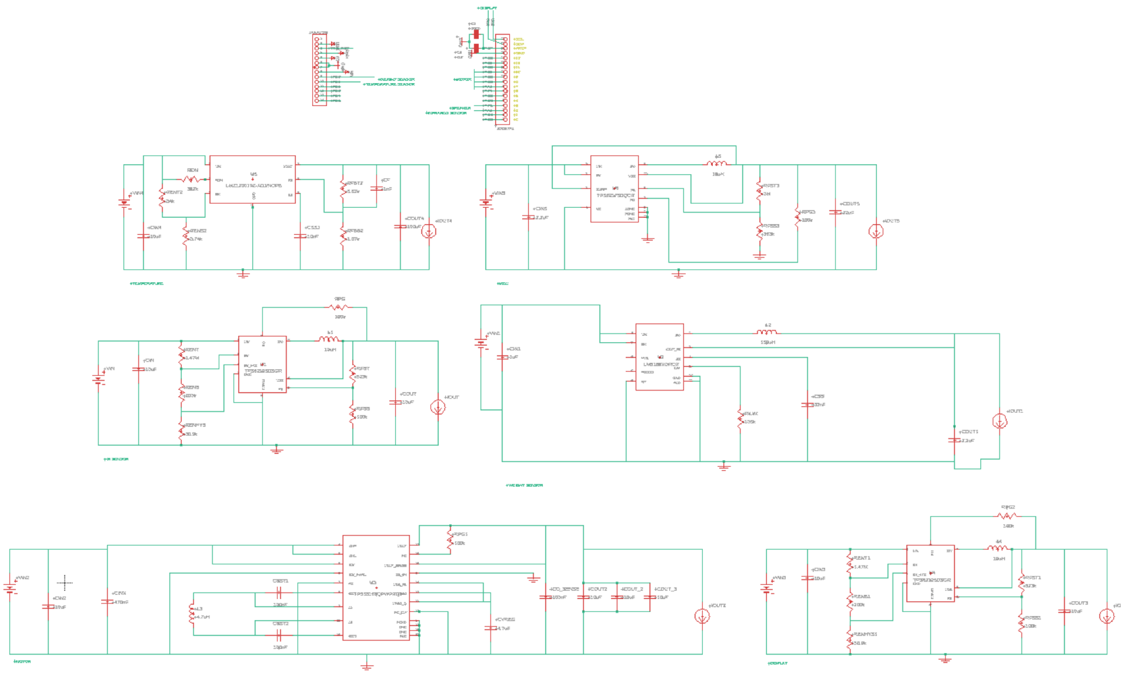


Figure 34: Voltage Regulator Circuit Connections - First Design

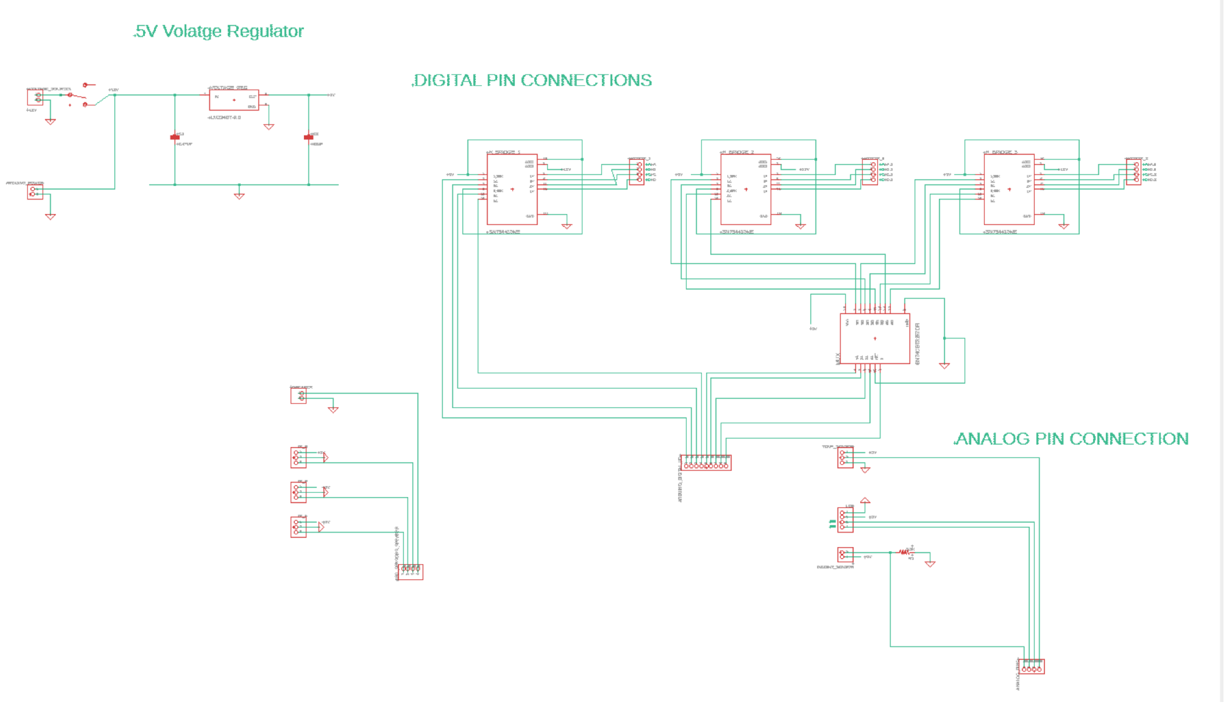


Figure 35: Circuit Connections with components - Second Design

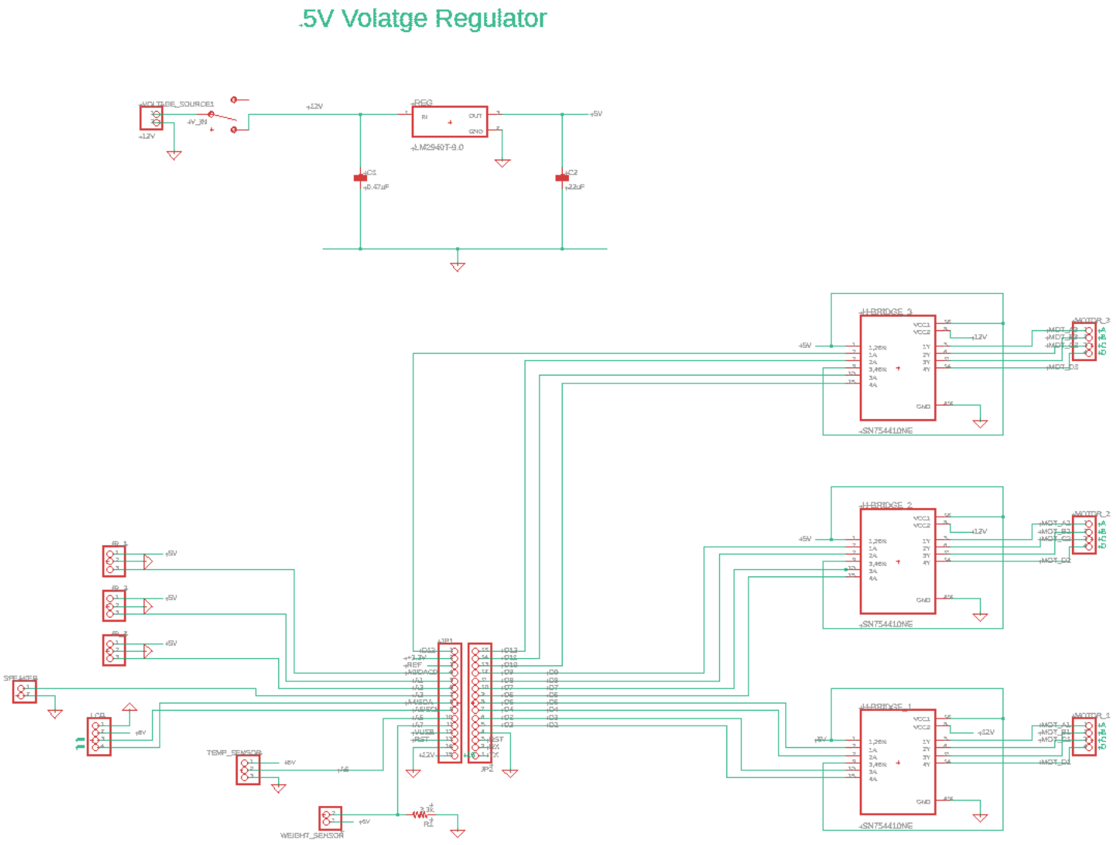


Figure 36: Circuit Connections with components - Final Design

As we can see in the figures above, as we kept on making changes we were able to obtain an enhanced design that will be later ordered and used for the final product. During this changing phase we encounter different suggestions that were later applied to get a successful prototype.

On the figure below we can see the first connections we made of the main components into the Arduino Uno Rev2 board. The motors take around 3 digital pins, the speaker takes one digital pin, and the IR sensor takes one digital pin. For the analog pins both the weight and temperature sensor takes one analog pin. As for the display it would take both the SCL and SDA pins in the digital pins of the Arduino board. This design was later changed, although some connections were still the same the MCU and the components changed.

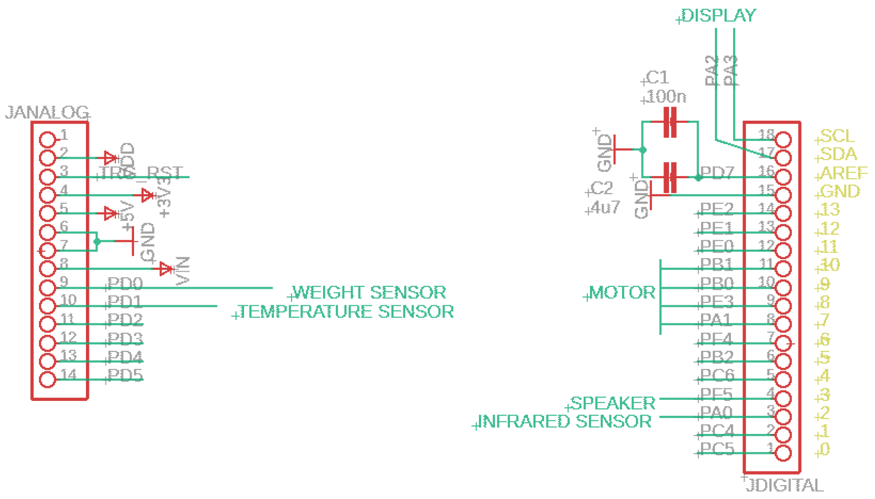


Figure 37: First Voltage Regulator Circuit Connections on Arduino board

In the figure below it shows the main MCU unit originally picked for the Pill[3] device, here we could see the pin layout for both analog and digital pins of the Arduino board, like it was previously mentioned the motors, the sensors and the display would be directly connected into the board using the PCB design to be implemented. This board was later changed to the Arduino Nano IoT 33, the design of it will be also shown below.

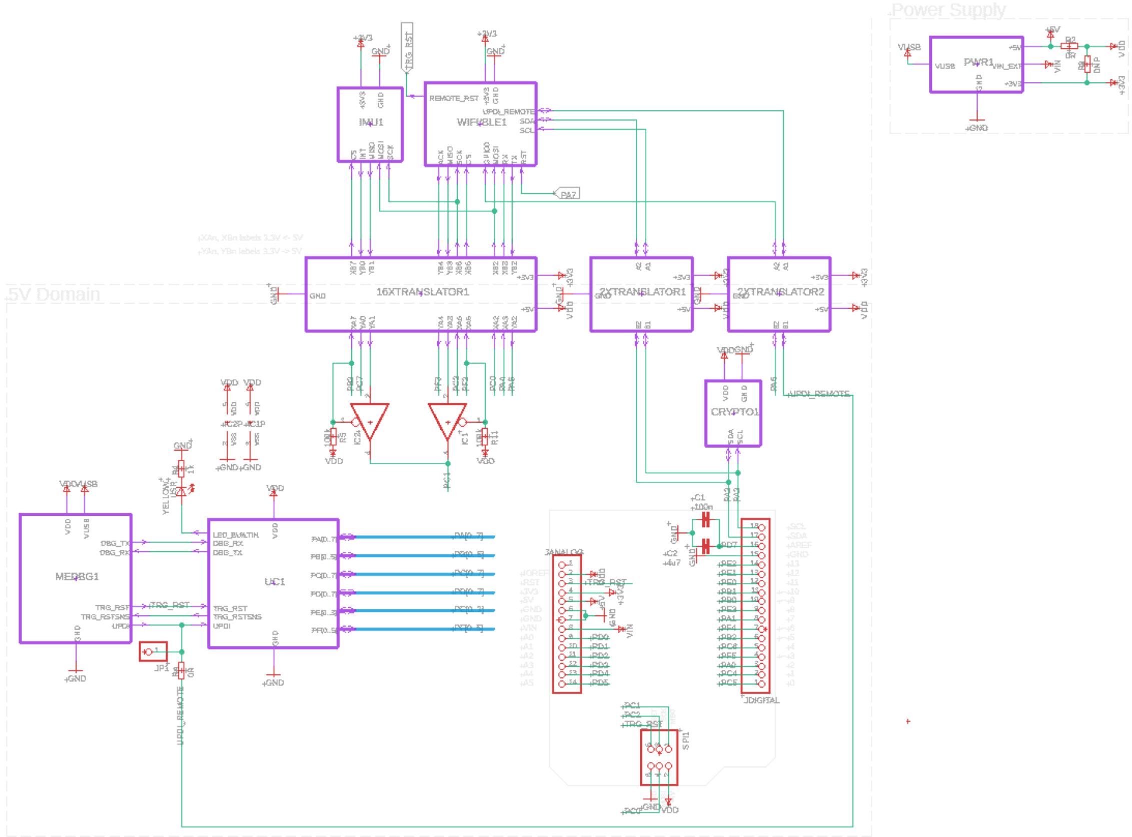


Figure 38 : First Microcontroller Unit Layout

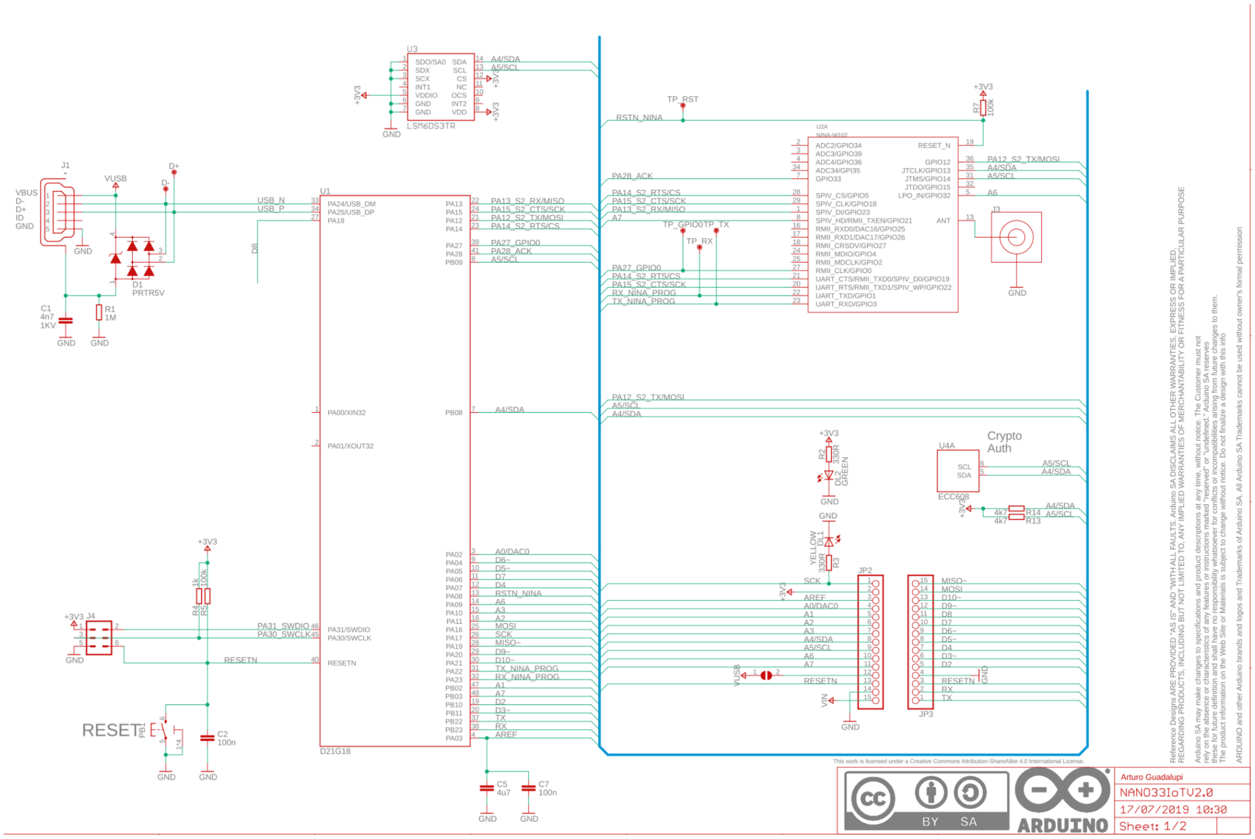

U3
LSM6DS3TR
A4/SDA
A5/SCL
+3V3
GND
J1
VBUS
D-
D+
ID
GND
VUSB
D1
PRTR5V
C1
4n7
1KV
R1
1M
U1
USB_N
USB_P
PA13_S2_RX/MISO
PA15_S2_CTS/SCK
PA12_S2_TX/MOSI
PA14_S2_RTS/CS
PA27_GPIO0
PA28_ACK
A5/SCL
A4/SDA
D21G18
TP_RST
RSTN_NINA
U2A
NINA-W102
RESET_N
PA12_S2_TX/MOSI
A6
J3
TP_GPIO0
TP_RX
TX_NINA_PROG
RX_NINA_PROG
+3V3
R2
330R
DL2
GREEN
U4A
Crypto
Auth
ECC608
SCL
SDA
R14
R13
4k7
YELLOW
DL1
R3
JP2
JP3
AREF
A0/DAC0
RESETN
VIN
VUSB
J4
PA31_SWDIO
PA30_SWCLK
R4
1k
R5
100k
RESET
PB1
C2
100n
C5
4u7
C7
100n
This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.
BY
SA
ARDUINO
Arturo Guadalupi
NANO33IoTV2.0
17/07/2019 10:30
Sheet: 1/2
Reference Designs ARE PROVIDED "AS IS" AND "WITH ALL FAULTS. Arduino SA DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE
ARDUINO and other Arduino brands and logos and Trademarks of Arduino SA. All Arduino SA Trademarks cannot be used without owner's formal permission

Figure 39: Arduino Nao IoT 33

Finally after multiple errors attained during the design phase we were able to finalize the design and implement it in the Pill^3. In the figures below we will see the two designs we sent to the vendors and were tested during the early stages of the Pill^3.

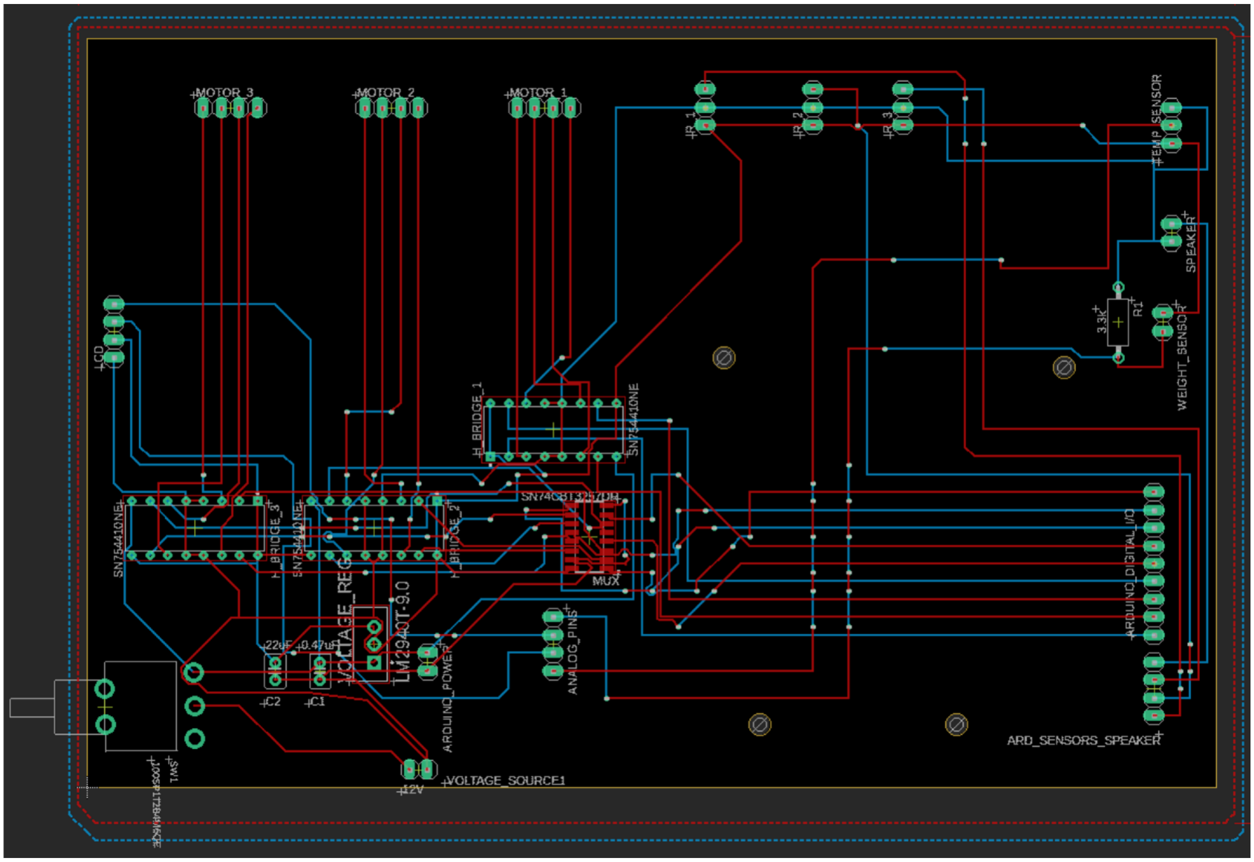


Figure 40: Printed Circuit Board - First Design

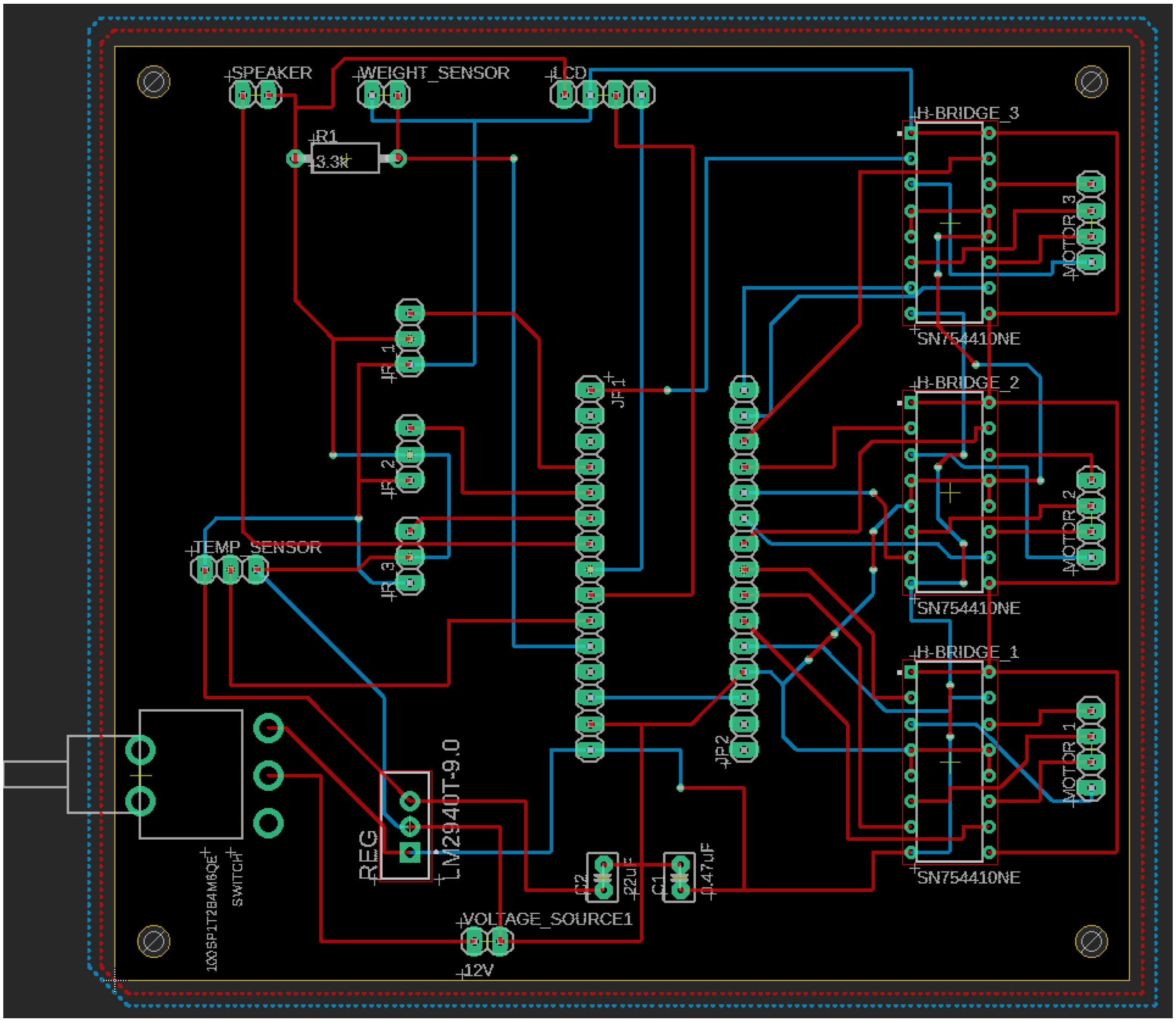


Figure 41: Printed Circuit Board - Final Design

8.1.3 Vendor

Once we have our PCB design finished and have met our set requirements, we start to look into vendors that could possibly manufacture our designs. We start to search for vendors that meet our standards. Aside from looking into vendors that have good reviews and are trusted by the community, we take into account certain other restraints that could affect us. This device was self funded, so having a limited budget plays a huge toll on us. The shipping and manufacturing time was also taken into account as we have limited time to finish our device and test our PCBs to see if it was functional.

We take our personal constraints into account and create a table to help us decide which company would meet our standards. Although each company holds its own merit, the table below shows how and why we decided to choose this PCB manufacturer opposed to its competitors.

Table 15: PCB Suppliers

---------------------	**PCBgogo**	**JLCPCB**	**SunStone Circuit**
Location	China	China	Mulino, Oregan
Cost	$120	$ 25	$ 102.18
Shipping	$ 26	$ 30.00	$ 58.00
Arrival Time	1 week	1 week	3 weeks
Total	$ 146	$ 55.00	$ 160.18

As shown in the table, JLCPCB would be our best option as it was the most cost efficient manufacturer on the market that could ship our design in a timely manner. We would be using this manufacturer for our device.

9.0 Project Prototype Testing Plan

This section of the documentation was dedicated to the setup and use of the testing environments, and discussion and analysis of the test results for each specific component of the Pill^3. This was used to determine the physical and electrical limits of each component and allowed for our team to account for these variables in the design and construction of the final product. Through these rigorous testing programs, our team was also able to verify information provided about the components in their datasheets. The program used to run the final product was also tested, to ensure that all connections and commands were properly implemented, and to determine if the hardware and software interacted as designed.

9.1 Hardware Test Environment

In order for the Pill^3 to be as successful and as sturdy as possible, our team has decided to test every single piece of hardware in our device thoroughly. We believe that doing so would leave less room for error and we might be able to gather more data on our device. We could observe what methods or components operate properly under certain circumstances and what doesn't. Aside from the hardware, our materials would also be tested such as the container to see the amount of stress that our device could support. Our goal and mission was to ensure that the Pill^3 was ready for anything. The environments and methods with which these components are to be tested would be explained.

9.1.1 Sensor Testing Environment

Since the Pill^3 was designed to use multiple types of sensors for different purposes, a different environment and set of tests must be determined for each individual sensor. However, certain benchmarks for each sensor need to be specified to ensure that they could properly collect the information required by the system to operate. Uniform benchmarks for the sensors include sensitivity, reaction speed, update time, and power usage. While the values of these benchmarks may vary from sensor to sensor, the importance of them does not. Therefore, in addition to any individualized tests for the sensors, these tests are included as well.

Infrared Sensors: To determine when a pill has passed a specified location on the dispensing mechanism and to accurately count the number of pills dispensed, IR sensors would be used. To accurately model the environment these sensors would be used in, the tests for these sensors would be conducted in varying levels of brightness, with more weight being lent to testing in darker surroundings. Since these sensors would be enclosed within the device, little to no external light should be present in their environment so it was essential to know how precise they are under those circumstances, and to observe any changes should any extraneous light enter the environment. In addition to the darker testing environment, the sensors would be tested with varying levels of reflectivity, distance, and orientation to determine the ideal placement for the IR sensor. The final test would be determining the sensitivity to changes in distance and reflectivity, which would be used to determine when an object passes this sensor.

Weight Sensor: The environment in which the weight sensor was planned to be used does not present any ancillary complications, so there was no need to create a special environment for this testing process. The sensitivity, weight range, and reaction time of this sensor would all be tested with no special conditions initially. After baseline measurements are recorded, the sensor would be tested for all of these specifications again, but with a plastic cup on the sensor at all times. This would be used to determine how precisely the sensor could detect changes in weight within the cup that would be caused by pills being in it. The sensor would also be tested to observe how it reacts to any impacts caused by pills falling several centimeters, to simulate what the actual dispersion method would be like.

Temperature Sensor: The final type of sensor used in the design of the Pill^3 was a temperature sensor which monitored the status of the operating environment within the frame of the device. It was important to be able to determine the temperature accurately, to within a few degrees, to ensure that neither the electrical components nor the medicine stored in the device overheated. Since the Pill^3 was a relatively closed system, aside from an air vent designed to increase circulation, the temperature sensor was tested within a closed box meant to simulate the inside of the device. It was then tested in several different locations within the box to determine if there was any specific location that yielded the most accurate data pertaining to the temperature of the box as a whole. When the location was finalized, its sensitivity was then tested for both slight changes in temperature of the whole box and drastic changes of temperature in one specific location of the box.

9.1.2 Motor Testing Environment

Only one type of motor was used within the design of the Pill^3, so only one set of tests was developed for this component. Any additional motors of the same type were tested for the same list of specifications. No special environment was needed to test the motors, so they were tested mounted to a wooden plank and supplied with power through a test microcontroller. The motor was physically tested for maximum speed, maximum weight able to be supported before performance degradation, and accuracy of control of rotation which included degrees of accuracy for turning to a specified location, starting speed, and stopping speed. The electrical performance of the motor was also measured to determine the optimal power input needed.

9.1.3 Speaker Testing Environment

Only one type of speaker was used for the User Notification System of the Pill^3, and the requirements of the design only required one speaker of that type, so only the speaker used for the device was subjected to this series of tests. As with the other electrical components in this device, the power consumption, voltage, and current of the speaker were tested for both its idle and active status. These measurements would be used in determining the power budget of the project as a whole. Besides electrical qualities, the speaker was tested for both frequency range and volume range, to determine what types of sounds are able to be played from the speaker and what ranges it was able to safely operate in for long periods of time. The distance at which the sound from the speaker could be heard at each volume was also measured, so that an accurate distance at which the user could be notified could be determined. The final test for the speaker was what types of materials could be placed over it without significantly degrading the performance of the speaker. It would be tested for no material, simple speaker material, and the material of the frame with either small holes or slotted lines.

9.1.4 Display Testing Environment

For the display, the testing criteria were uniquely centered around Graphics User Interface (GUI) readability and clarity. As with other components, the power consumption was tested in both idle and active configurations, and the speed at which the display could be updated was examined. The status of all pixels was tested with different solid color screens, to determine if any pixel was broken or had difficulty displaying a certain color. Various GUI setups would then be tested for both the clarity of information given, the brightness of the screen, and the distance at which the text on the screen could be read. Our team wanted to determine accurately how a user would be able to gather information from this screen, and thus focused our testing of this specific device on that.

9.1.5 Power Supply Testing Environment

To ensure that the power supply was working as intended, its electrical properties were tested thoroughly. After the device was connected to a wall socket, the input and output voltage and current were tested every hour for 24 hours to monitor any changes in power delivered due to constant use. The temperature of the power supply was also measured during this period to determine if there was any significant change in temperature to the environment due to it. Although the power supply was planned to be external, the change in temperature caused by it needs to be determined to both ensure that it isn't a fire hazard and to predict if it may cause changes to the internal temperature of the device.

9.2 Hardware Specific Testing

The actual testing for each component was carried out as specified in the Hardware Test Environment section of this document. Before extensive specialized hardware testing begins, simple testing was done to ensure that the components function properly, to plan the layout for the component circuits, and to calibrate the components and programs for typical use. This section would record and analyze the information gathered while testing and any changes to design required by the outcomes of these tests would be discussed here.

9.2.1 Infrared Sensor

The first sensor analyzed was the infrared sensor used to track and confirm the distribution and number of pills. This sensor transmits a simple, digital binary signal to the microcontroller of either true or false. If the IR light released from the black diode was absorbed by a black material or otherwise failed to reflect back to the clear sensing diode, the input to the microcontroller would read "1" or true. When an object was in front of the sensor, the IR light emitted was reflected back to the clear sensing diode, and the input to the microcontroller was read as "0", or false. While this may seem unintuitive, as one would typically use "1" to inform a microcontroller that the sensor was detecting something rather than the absence of that something, unfortunately the input values were determined by the vendor of this component and could not be easily changed by the team. However, this should pose no problem as the program utilizing the IR sensor would be written with this in mind.

The wiring configuration during initial testing for this component was simple, consisting of a voltage connection, a ground connection, and a data transfer connection. This sensor requires an input voltage of 3 to 5 volts, similar to the other sensor used by the Pill[3]. For the initial testing of this sensor it was connected directly to the 5 volt and ground ports of the arduino microcontroller, and the data transfer wire was connected to the digital port 3. When the sensor was properly connected and power was supplied to the arduino, a single green LED on the IR sensor was lit, signaling that it was operating correctly. When the sensor was moved in front of an object and the IR light was reflected, a second green LED would light up, indicating that it was functioning as intended.

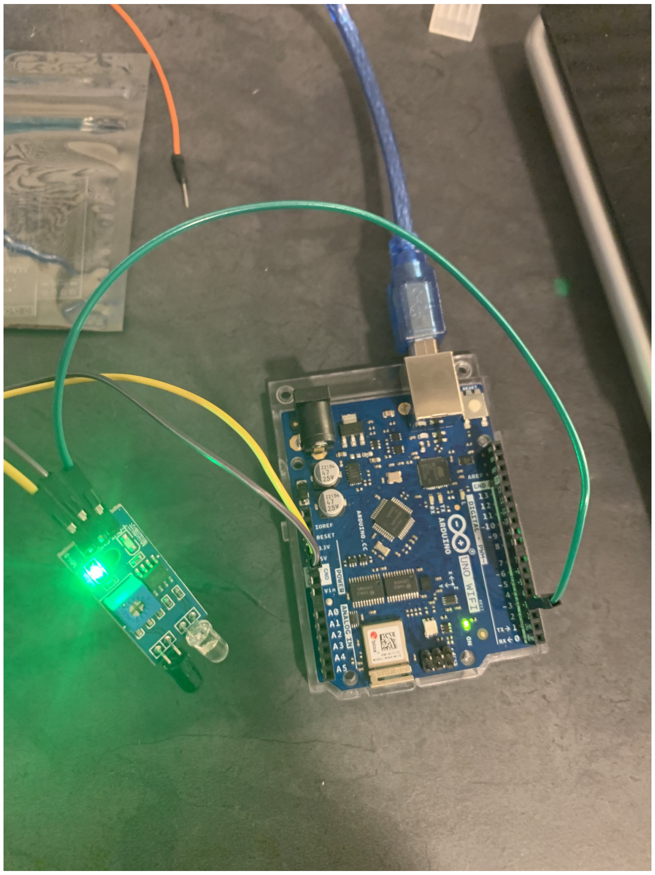


Figure 42: IR Sensor Initial Test Environment

To test the logical input of the IR sensor a simple program was uploaded to the Arduino, defining digital port 3 as an input. The value was then read from that port and printed to a command console with a delay of 1 second between each reading. When the IR sensor was pointed at black electrical tape, the same material that our team plans to use paired with this IR sensor in the device, the expected reading of “1” was outputted to the console. When an object was passed in front of the IR sensor, the output would change to “0” until the object was moved out of the line of sight of the IR sensor, at which point the output would return to “1”. This test was repeated several times, and at every occasion the output would change to the expected value within the 1 second delay. The exact response time of the IR sensor has not yet been measured, however there should be no need for it to respond faster than 1 second as long as the speed of the motor was set with this time frame in mind. Since this component worked as expected and the wiring, programming, and calibration of it was complete, initial testing of the IR sensor concluded with these findings.

9.2.2 Weight Sensor

The weight sensor component used in the Pill^3 operates like a variable resistor, with the increasing weight changing the value of the resistance and thus affecting the voltage going through it. In the initial setup used to test, the weight sensor was connected to an analog pin on the arduino microcontroller, and sent an analog signal containing a voltage level in millivolts. By using equations provided in the components data sheet, the measured voltage could be converted into grams. Since this was an analog value rather than a digital logic value like the one used in the IR sensor, the accuracy and content of this sensor must be verified during the initial tests.

The wiring configuration during the initial testing of the weight sensor included a simple RC circuit constructed between the sensor and the input voltage. A 0.1 uF capacitor was placed in parallel with a 1 MΩ resistor, with one end connected to the 5 volt pin of the arduino and the other end connected to the ground pin. At the same node as the resistor, capacitor, and voltage pin, the right side of the weight sensor was connected. The left side of the weight sensor was connected to the analog pin A0 on the arduino microcontroller, with no other components attached to it. Although the weight sensor itself did not have any physical indication that it was connected properly, when pressure was applied to the circular end of the sensor the console registered a change in voltage. When no force was being applied to the end of the sensor, the metal material in the sensor does not touch, creating an open circuit and thus producing an input reading of 0 volts. In the image of the initial weight sensor testing, the stepper motor was placed on this circular end to calibrate the accuracy of the weight sensor.

The output data for this type of sensor was an analog voltage reading which can, using certain conversion equations provided in the components data sheet, be translated into the weight of an object in grams. To perform this the input must first be converted from an analog signal to a digital one by multiplying the signal by the input voltage of 5 volts and dividing by 1024, as the Arduino microcontroller analog pins use a 10 bit analog to digital converter. Because of the linear relationship of the value of the resistance of the sensor to the amount of force exerted on the sensor by an object's weight, the weight of an object in grams could be calculated by multiplying the measured voltage by a calibration factor. This calibration factor was determined by taking the weights of several objects measured on a digital kitchen scale and adjusting the value of the calibration factor until the weights remained consistent between the two measuring devices. By comparing these objects the initial value of the calibration factor was set to 195, which produced weights within 10% of the actual measured value of the object. While this was a relatively high amount of error, in the context of the weight sensor it shouldn’t hinder its intended functionality. The sensor only needed to detect when a cup weighing more than 20 grams was present, as long as the measured weight was above or below this threshold, regardless of the exact value, the device would be able to function as intended. As such, initial testing for this component was concluded with this assumption.

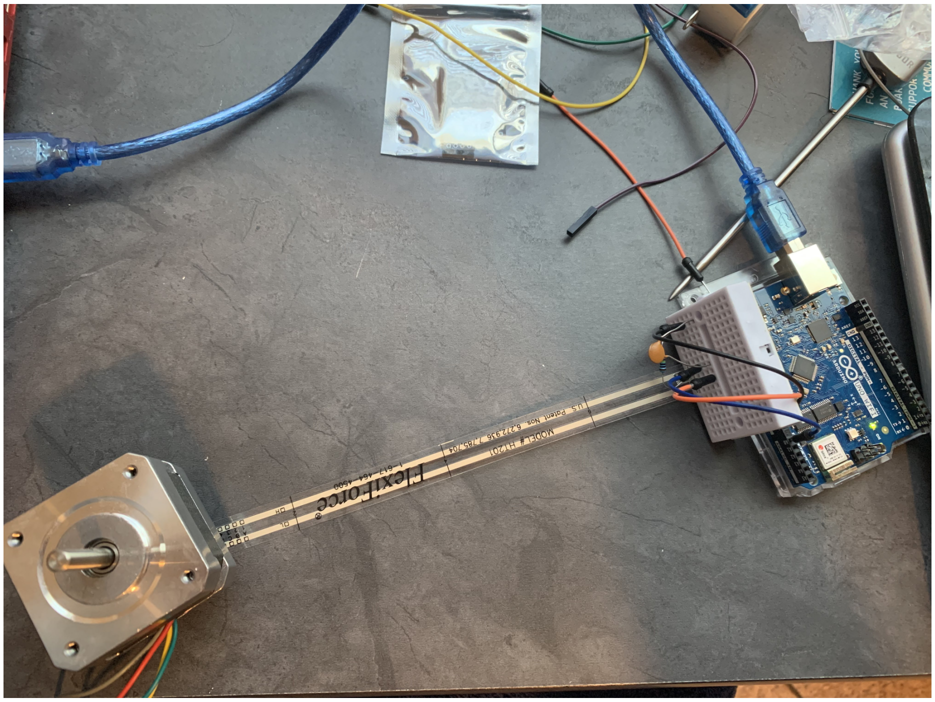


Figure 43: Weight Sensor Initial Test Environment

9.2.3 Temperature Sensor

The temperature sensor used for the Pill^3 operates on the simple relationship between temperature and diode voltage. The sensor itself was little more than a diode in an enclosure that changes its output on a predictable algebraic equation of voltage and temperature. For the initial testing set up, the temperature sensor was connected to the microcontroller through a 5 volt input pin, an electric ground, and an analog input pin. The sensor sends an analog voltage signal to the input pin that could be converted to a temperature value through an equation provided in the component data sheet. For this project, our team has decided that the accuracy of this sensor needs to be within 2 degrees fahrenheit to properly ensure that the medicine was stored well within an acceptable temperature range, so the initial tests for this sensor focused heavily on accuracy and time response of the temperature sensor.

The wiring configuration used for the initial testing was a simple connection between the 5 volt pin, the electrical ground pin, and the A1 analog pin of the microcontroller. This sensor could operate on a range between 3 and 5 volts, however our team decided to use 5 volts for this sensor as standardizing the voltage used by all the sensors simplifies the design of the power supply circuit and reduces the amount of electrical components needed to construct it. Similar to the weight sensor, the temperature sensor had no external indications that it was operating, so a simple program was written to read the inputs of this sensor and display them on the debugging monitor. The temperature sensor was shown to be connected and working properly, as a stream of analog readings were observed through the console.

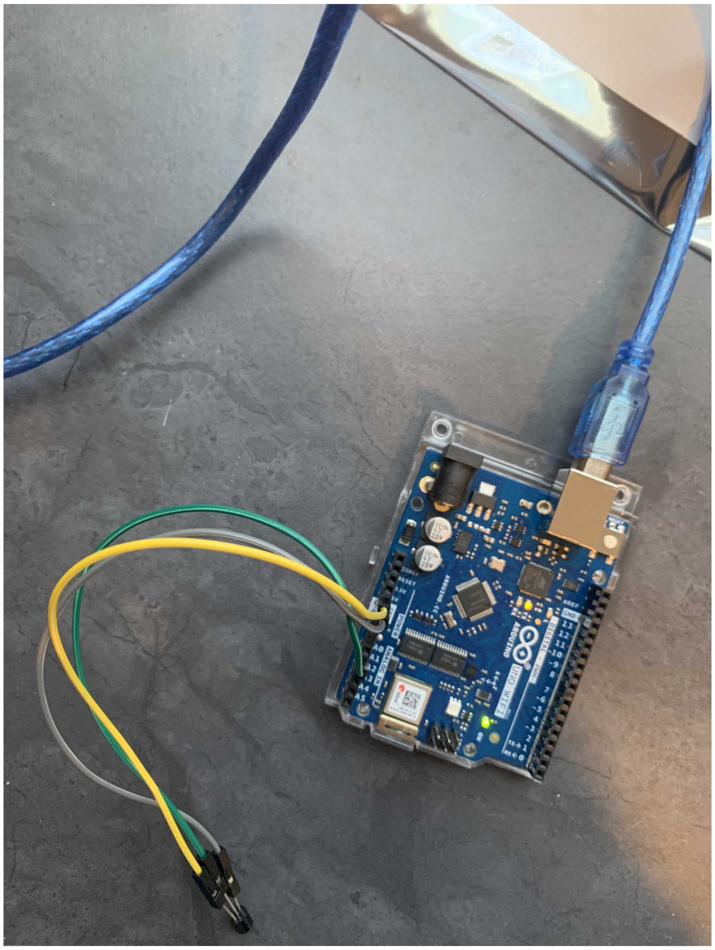


Figure 44: Temperature Sensor Initial Test Environment

Similarly to the weight sensor, the temperature sensor provides the Arduino analog pin A1 with a voltage signal indicative of what the sensor was measuring. In order to be able to understand and display this data, it must first be converted to a digital signal by multiplying the signal by the input voltage of 5 volts and dividing by 1024, as the Arduino microcontroller analog pins use a 10 bit analog to digital converter. To parse this digital signal into a readable temperature, the data sheet of the temperature sensor component was used to determine the correct equation to use. First an offset value based on the voltage out equivalent of the lower end of the desirable temperature range and the voltage out equivalent of the higher end of the temperature range multiplied by the ratio of the output voltage range and the desired temperature range was subtracted from the measured and converted temperature. The next step of conversion was to divide the resulting value by the ratio of the output voltage range and the temperature range which results in the temperature value in celsius. For convenience, and because this device was primarily intended for use in American homes which typically use fahrenheit in temperature displays, the temperature was converted from celsius to fahrenheit by multiplying the celsius value by 1.8 and adding 32 to the resulting value.

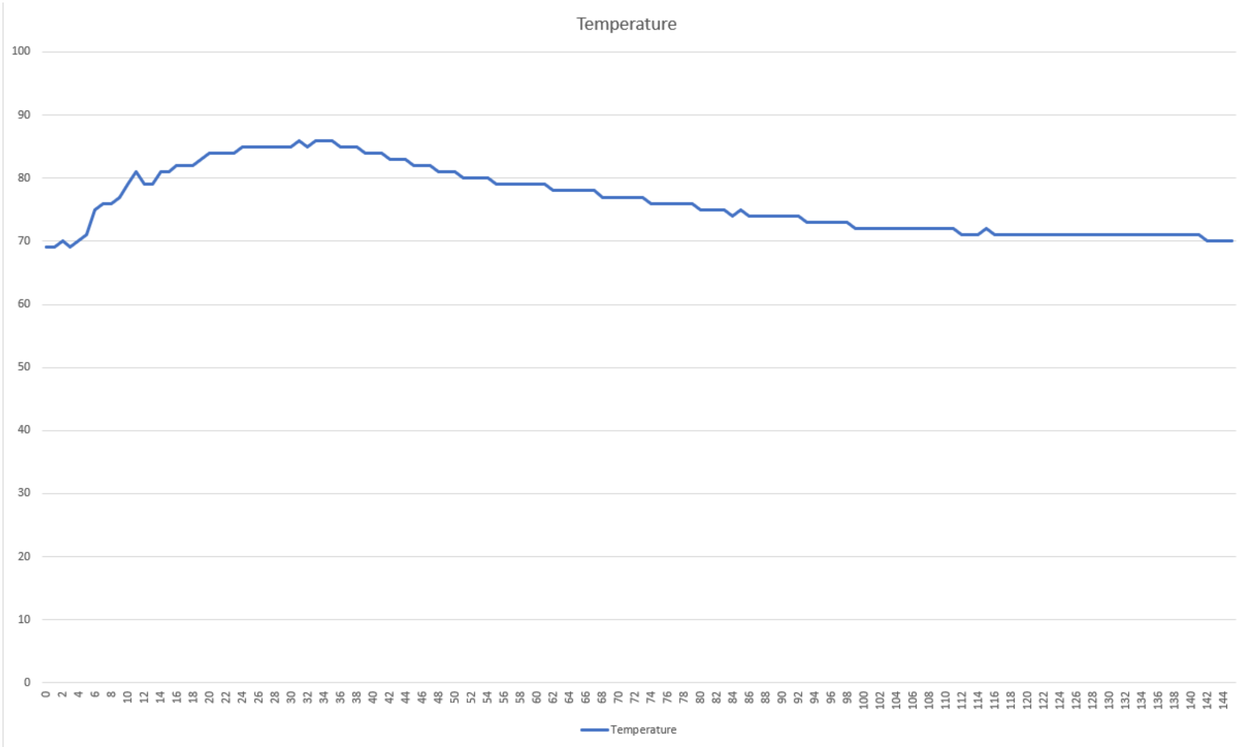


Figure 45: Output of Temperature Sensor

The graph in figure 39 shows the output of the temperature sensor over a range of two minutes and 25 seconds. The initial starting room temperature of 69 to 70 degrees farenheit was confirmed by both the air conditioning unit of the room which was set to 70 degrees fahrenheit, and a digital meat thermometer which read the same. The sensor was then held in the fist of team member Liam Kenney to increase its temperature to measure the reaction time and the maximum temperature reached. Over the course of 30 seconds the temperature increased to a peak of 86 degrees fahrenheit, the same temperature measured by the digital meat thermometer. After the peak temperature was reached, the sensor was removed from the fist and placed back in an open location to return to room temperature. The return to room temperature took approximately 1 minute and 55 seconds before the sensor stabilized around 69 to 70 degrees once again. While initially this long return period may seem concerning, it was actually entirely expected due to the way heat was transferred to and from the sensor. Since this sensor would be used to measure and monitor slight changes in environment temperatures over long periods of time, this delay time would pose no issue for the sensor or the Pill^3 system. With this conclusion, the initial testing of the temperature sensor was ended and the configuration settings were saved for later use.

9.2.4 Motors

The adafruit stepper motors that our team plans to use for the Pill^3 storage containers and distribution have 4 connection wires, two for power and ground and two for data input and control. In order to properly power and control this motor, an H-Bridge chip would be used as an intermediary between the stepper motors and the Arduino Uno microcontroller board. This chip would supply logic commands, power, and electrical ground to the motor and would also include a heat sink to ensure that the connection runs smoothly. The exact model number of the chip that our team plans to use was the SN754410 H-bridge motor control chip, however at the time of component testing and report writing, this chip was not available. As such even though the stepper motor was obtained, initial testing for the motor was unable to proceed until this chip arrived.

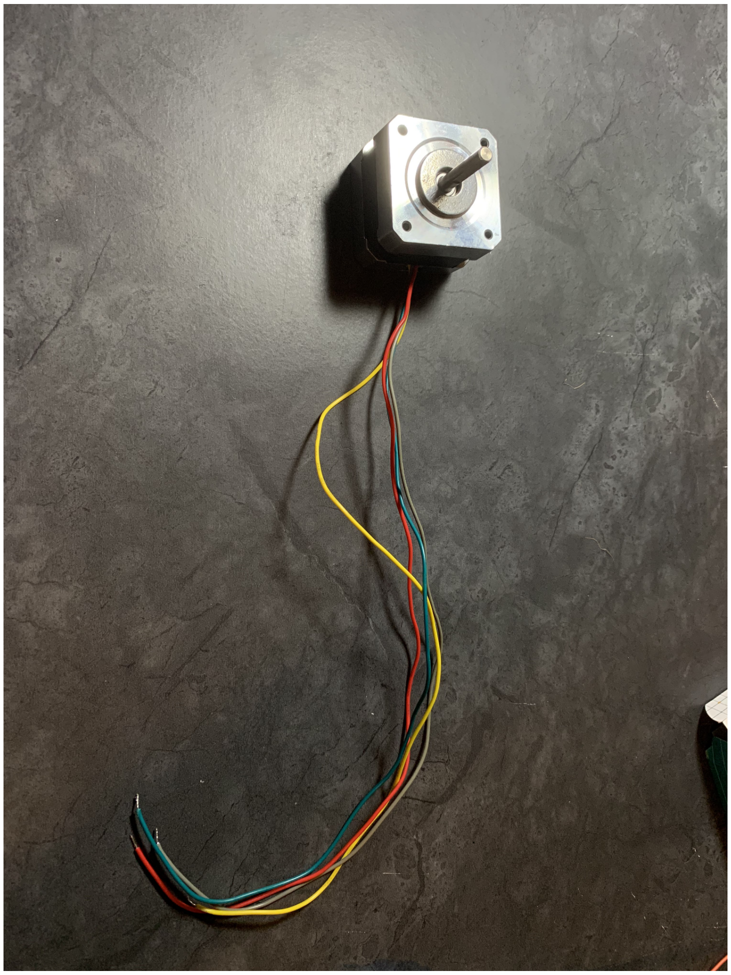


Figure 46: Stepper Motor

9.2.5 Speaker

The CQRobot speaker, chosen by the team to alert the user of the Pill^3 to any status changes, was electrically simple and could be driven from a digital output pin of the Arduino microcontroller. It has two wires, one which requires an input signal and the other a connection to an electrical ground. Based on the frequency, duty cycle, and power of the signal inputted into the speaker the volume, tone, and pitch of the sound was changed. The speaker itself consists of little more than a piezoresistive material that is vibrated according to the programmed tone. Since our team decided that the speaker need only alert the user to an ongoing issue, not explain or specify that issue, we decided against using an audio driver that allows for complex sound generation. Instead, as pictured in the figure below, the speaker was connected directly to the Arduino board through a digital pin and would be driven by a pulse-width modulated signal.

When connected as shown, a program was run that played the "B" note at maximum volume to ensure that the speaker worked as intended, and to gauge the range at which it could be heard. The device was tested in the apartment of one group member to mimic a typical household environment as accurately as possible. The speaker was then tested under two circumstances; relative silence where no household activities were being performed, and a simulated "active" environment, where noise from an individual cooking was introduced into the environment. In both tests, the speaker was able to be heard clearly throughout the environment, even through two or more walls and closed doors. The "active" environmental sounds were no impedance to the user being able to hear the speaker, even when they were in a different room or attending to the work of cooking themselves. The volume level and exact tone of the speaker would be later adjusted to a more natural, less startling noise, however the current test procedure proved the usefulness and range desired from the speaker. As such, initial testing was concluded and the settings for the speaker were recorded.

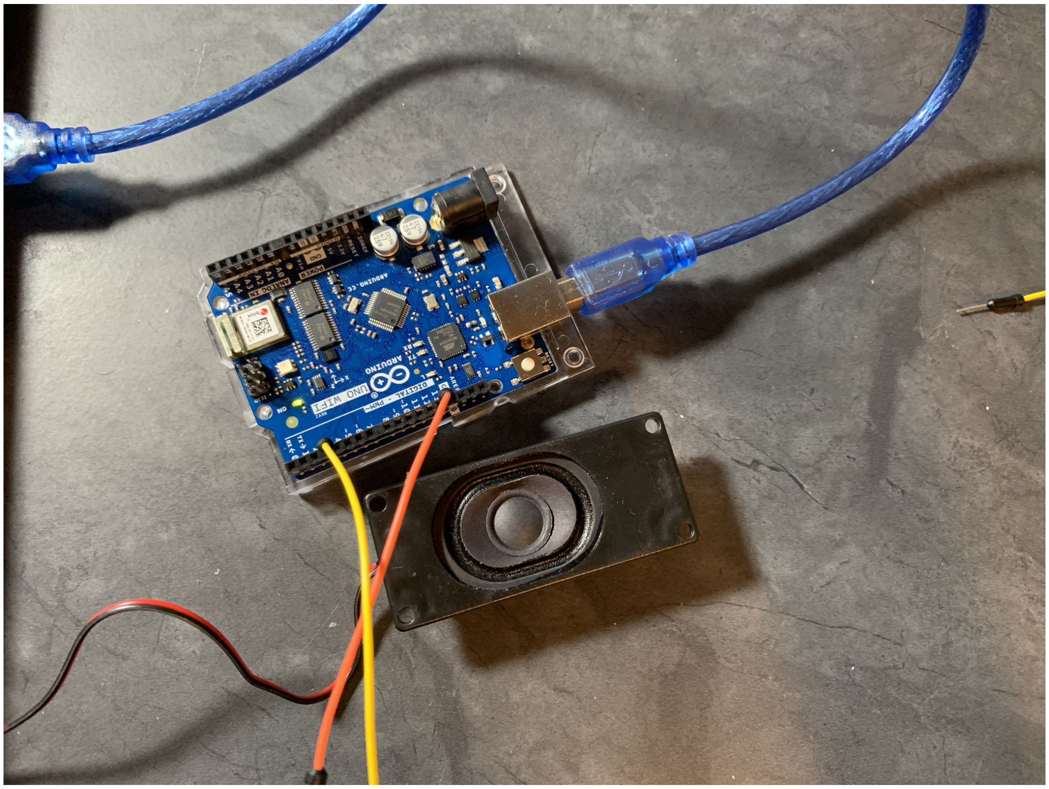


Figure 47: Speaker Initial Test Environment

9.2.6 Display

The OLED display chosen by the team was perhaps the most complicated component program-wise, although electrically its connection and operation are quite simple. Lines of light emitting diodes are organized to create rows and columns of pixels, which could each be controlled individually from the Arduino microcontroller. For a simple small display such as this, no additional video drives are needed, and it could be run straight from the Arduino board itself, similar to the speakers used in this project. Additionally, since this display was only intended for use as a secondary source for device status information, it was deliberately chosen with simplicity of use and design in mind. The OLEDS inside the display, aside from being extremely power efficient by turning off when a black pixel was desired, are capable of producing either plain white light or a blue-ish colored light. Utilizing these 3 colors, the display was planned to be able to display the time, the strength of a wifi connection, and have multiple indicators for different device states, which would be explained through the user manual.

At the end of the fall 2021 semester, the OLED display was received; however it was unable to be tested as it did not come soldered with pins, and the material and tools to solder input/output pins were not readily accessible before the end of the first semester. As such, while the program and wiring diagram were able to be planned out, the actual component was not able to be tested yet until the start of the senior design 2 in the following semester. We were able to utilize the program and wiring diagram as planned.

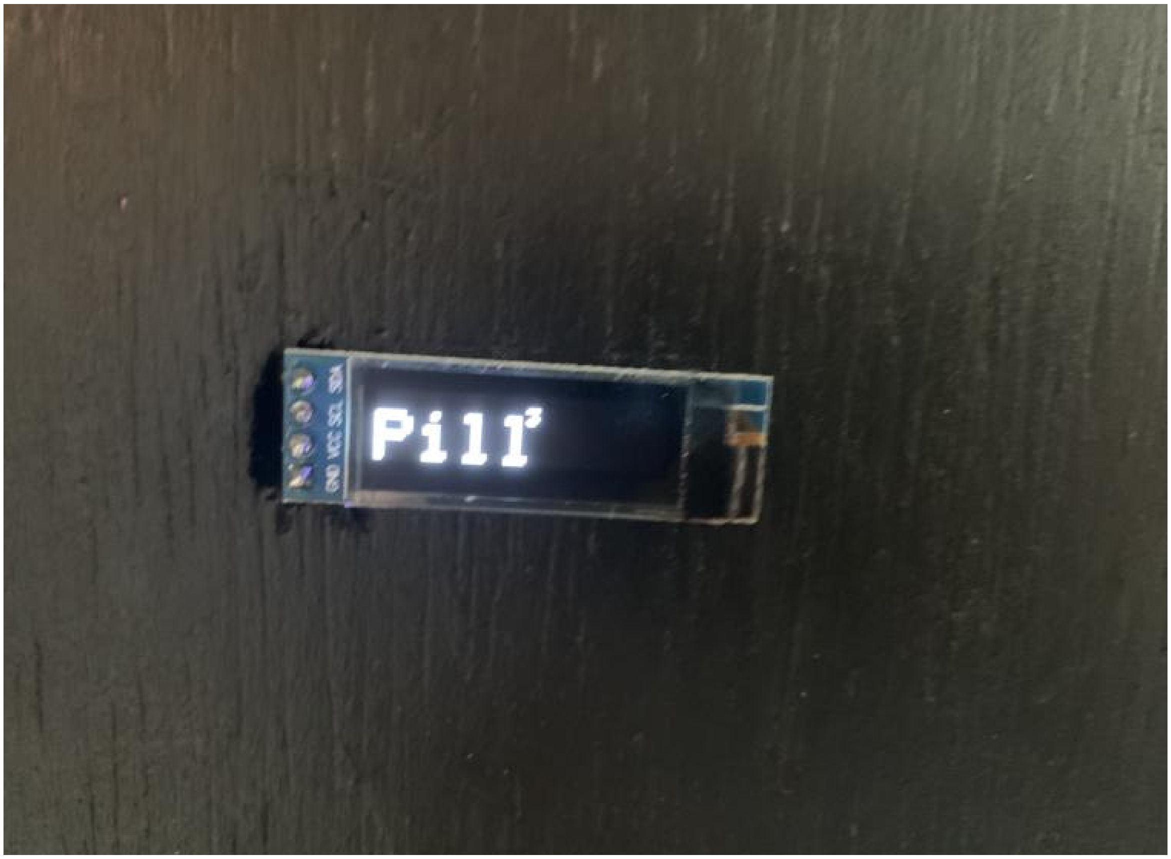


Figure 48: OLED Display

9.2.7 Power Supply

The Pill^3 would be powered via a wall outlet. We would be using an AC/DC converter in order to do so. The power supply was one of the most important parts of our device. It was what kept the Pill^3 alive and what could have the potential to ruin it based on the power given. In order to test our AC/DC converter, we would plug in our power supply to an outlet and test it out with a multimeter. With the multimeter being set to read DC, we plug the positive end of our power supply to the red probe of the multimeter and the black probe to the barrel of the plug. Based on our readings, we should be able to figure out if it was working correctly. If the reading was negative, the probes need to be switched. By doing so, we could see how steady our power supply is.

9.2.8 Chassis

The chassis was the body of our device. It was what carries every piece of hardware and protects it from outside factors. Although it was not a piece of electronic hardware, it was important to test the durability on the chassis and try different materials to see what was best for our device. We would be testing three different types of materials. Plywood, acrylic and filament for a 3D printer. We would be putting it under a number test to see how much weight it could withstand, and its resistance to wind, dirt and water. The purpose of these exams are to see how secure the device was when put under certain conditions.

9.3 Software Testing

For testing software there are a large number of choices available; however, as the app gets to a larger scale it becomes gradually easier to keep testing systems. With this, it was also more useful on larger systems to have testing as more foundational things could start to get too complicated to keep track of. We have already decided on a platform to use (mobile) along with a framework (React Native), a database to use (firebase), and a backend (Node.js)- we could then use this information to decide on the best software testing practices and what kind of testing we plan on using.
Something to keep in mind was readability. With testing, it was imperative to have legible tests. Without this, testing could become a huge nuisance and become a hindrance to anything that could change down the project's path. ***With this stated, it was extremely important to consider the look of the code, so there would be code snippets shown below to determine legibility.***

One of the biggest things to consider when testing was the use-case of each testing framework or strategy. In this case, one of the primary things to consider was the fact that this app was a small-scale app so it may not require the biggest, complicated, in-depth framework. A few things to consider are:

- *Ease of Use*
  We want the unit testing to be as simple as possible. The point of unit testing should not require any complex calculations, struggles, or overly-complicated functionality.
- *Support*
  Support for anything was always something to look out for. Without support, solving issues that are bound to come up may become more and more difficult.
- *Applicable to React Native/Firebase/Node.js Stack*
  We need the testing framework to be able to be used with the technologies that are being used. Although a testing framework may be something that was extremely popular within the community, if it doesn't work with the planned technology it can't be used.
- *Readability of Code*
  For unit testing, readability of code was important for other people to look over your code. Although this was something that was necessary in most programming languages,

having readability in unit tests make debugging testing functions easier. Being that testing was what determines if your code even works, it was of the utmost importance that testing was written correctly and quickly so it doesn't become a hindrance.

9.4 Frontend Testing

Frontend testing covers how your frontend looks on mobile devices and desktop; as we are creating a mobile application, we are looking for a way to test frontend components. Jest, Detox, and manual testing are popular choices in the industry used in front-end unit testing.

9.4.1 Jest

Jest is a unit testing framework that works with both the frontend and backend. Their frontend testing utilizes "snapshotting" that allows a user to specify a "screenshot" of the component and compare them to an object with specific presets. This covers cases where a UI may change unexpectedly.

Jest was made by Meta for React and React Native in mind. There was a ton of support for React and React Native as Jest was what React developers at Facebook use and Jest was directly built-in to React Native so setup was very clean and simple. However, using Jest requires a big foundation which may not be exactly what we are looking for in regards to frontend testing for this project.

There are some upsides to using Jest for this, Jest's snapshots work on any serializable object which allow a user to screenshot any object in any language and capture it to make sure the output was as the developer expects. This was called snapshot testing, which in regards to other forms of testing called visual regression testing where they screenshot the actual screen and compare the output pixel-for-pixel. Jest is much more flexible in terms of testing components and makes it a great choice for projects with lots of components.

Readability of Jest was not very good. Although someone could spend time to break it down, there are some syntactical choices that make Jest easier to read than some other frameworks. If you consider a component to check called "comp" there are a few hoops to jump through:

```
const comp = renderer.create(<Component />).toJSON();
expect(comp).toMatchSnapshot();
```

It was not exactly clear what was happening until you cover the `<Component />` tag, then converting to JSON, then matching snapshot, which then has to already have a pre-copied snapshot of the working component. With this, we could then compare it to the new component we are creating. For testing, there should be minimal obstacles- for a smaller scale project such as this, we want to minimize these obstacles to maximize efficiency.

There was one big thing that deters our project away from Jest for frontend testing which was that the testing was used to compare outputs, and does not apply test-driven development principles. Jest frontend would compare a new output to older outputs and make it easier to see changes between components over their life; however, this was not exactly what we need. We are looking more specifically for a way of seeing possible conflicts with our frontend UI- with only one person in charge of the software portion of this project, it was unnecessary to have a way of keeping track of component's changes.

9.4.2 Detox

Detox was a frontend testing framework that was developed by Wix. It was an E2E (end-to-end) testing framework which means it runs your app on an actual simulator and interacts with it just like a user would. It works with React Native as well as native code such as SwiftUI or Java for iPhone and Android, respectively.

The good thing about Detox compared to Jest was the simplicity. Although Jest's readability was great and Detox's isn't, Detox's logic makes just as much, if not more, sense. This was due to the reference of a "Device" object. You could write tests as you would a user by functions such as `device.tap(), device.typeText(), and device.tapReturnKey()`. In terms of testing, this would simulate a user as close as manual testing.

The support for Detox was minimal as usually Jest was considered the industry standard for React Native, so most companies choose Jest as it works well for large-scale. Detox was a testing framework only a few years old and not adopted by many companies.

In comparison to Jest, there are a lot more moving parts in Detox. This was a major downside as for a project of a smaller scale such as ours, we want something with little error margin. This makes Detox less viable as a solution. The setup for Detox could get very complicated if not configured perfectly, which leaves a lot of room for error. As Detox does simulate a mock user on a simulated device, if everything isn't set up just right the mock user would not do its testing actions, which on a small scale renders Detox slightly less useful than just manual testing at that point.

9.4.3 Manual

Manual testing was something for larger apps that was extremely frowned-upon. On smaller scales, manual testing for frontend may be the better option due to not having to install testing frameworks, smaller margins of error, and maximizing time efficiency.

Support for manual testing was something that doesn't even have to be considered as it was on an individual project basis. It is applicable to any technology stack and does not require any code.

The main thing with manual testing was having to keep track of all of the exact requirements to test. This requires a list of manual tests so that it could be repeated by anyone who wishes to test it. There are a slew of formats to use for manual testing, the most common one being a simple numbered list for anyone to easily follow along and replicate the test.

Summary

For our project, manual testing for frontend was most likely the most viable option. As the frontend isn't going to have a major platform, lots of moving parts, and only a few pages, manual testing would be minimal and something that does not look right could be easily caught by our eyes. Front-end testing for larger projects are almost always done with frameworks as the amount of things that could go wrong in a large codebase are much greater.

This, along with the time efficiency of not using a frontend testing framework, makes manual testing the best choice. If time permits, a frontend framework would be chosen and used to further validate the process of modifying front-end features.

Table 16: Manual

Testing Framework	**Ease of Use**	**Support**	**Applicable**	**Readability**
Jest	3/5	5/5	Yes	2/5
Detox	3/5	1/5	Yes	3/5
Manual	5/5	N/A	Yes	5/5

9.5 Backend Testing

Backend testing requires testing the connection between, in this case, the connection between the application and the database. In this case we are using Firebase, so we need a simple, legible, and lightweight backend testing framework. Unlike frontend, you can't manually test backend due to backend being something a user could not directly interact with.

For our backend it would be written in javascript, as that easily connects with React Native. Thus, we are looking for a javascript unit testing framework to be able to mock responses and make sure our functions that are calling to the backend are working efficiently and correctly.

9.5.1 Jest

Although Jest was covered in frontend testing, it could also be used for backend testing. Just as previously stated, Jest was built-in to React Native for easy setup- all it required was an import to a file where the tests would be written. Jest works out-of-the-box and there was little to no setup required.

Jest was a testing framework made by Meta and was an industry standard for all-around unit testing. Even for smaller-scale projects, having unit tests was extremely important; Jest makes writing backend tests really simple by having built-in code coverage tests, running tests quickly in parallel for time efficiency, and a simple API.

Although snapshotting for frontend testing with Jest could get complicated and was harder to decipher, unit testing backend javascript functions with Jest was simple and extremely legible to developers. The API has functions that are built on method-chaining and read as if a sentence:

```
    expect(value).not.toBe(50);
```

This makes creating unit tests very simple with little margin of error.
Because Jest was an industry standard and a big part of unit testing frameworks, there was a ton of support for it online and it fits in perfectly with everything small- or large-scale.

9.5.2 Mocha

Mochajs was another popular option; however, not nearly as popular as Jest. The set up of Mocha was also a little more complicated than Jest, which was built-in. This was due to Mocha not coming with mocking, assertions, or stubbing libraries. Instead, it requires a third-party choice of another library or framework. This was a huge disadvantage when it comes to lightweight projects. Although this was something that Mocha and other reviewers considered "flexible", for this project it was a downside.

Support for Mocha was pretty widespread. It works with a lot of technologies and it was adopted by enough tech companies that it was also an industry standard. There are plenty of documentation and API docs for Mocha. Having a dependency on a third-party library also requires more documentation to sift through if any errors arise, along with having to worry about compatibility with many different libraries.

Because of Mocha requiring a third-party library/framework, it was harder to directly compare readability. Mocha reads very similar to Jest but doesn't have the features Jest has. It reads very similar to Jest:

```
expect(numbers).to.be.an('array').that.includes(2);
```

This code snippet was used with Mocha.js and Chai.js.
With the level we plan on using a testing framework, having assertions, mocks, and stubs built-in so that our project doesn't require a bunch of different dependencies. Mocha's readability was great, but without those features, it was harder to look at more than the vanilla features' readability.
Using Mocha was simple, however with the excessive method chaining it could start to become more difficult on how to split it up compared to Jest. Also due to the dependency Mocha requires, there could be a lot of overlap on what library was using what. This leads to many ways to do the same thing as Mocha doesn't nicely package everything up for developers to easily write unit tests. In comparison to the example above, it was using the "Expect" style of unit testing, where there are at least 2 other ways to do this same exact thing:

Assertion: `assert.include(numbers, 2, 'array contains 2');`
Should: `numbers.should.be.an('array').that.includes(2);`

Although many ways of doing something may be easier to read, having different options provide more room for error, and also further complications when mixing and matching. This along with requiring another library could get even more complex as more functionality was necessary for unit testing.

9.5.3 Jasmine

Jasmine was one of the other top javascript testing frameworks used in the industry. It was mainly used with Angular as it came shipped with Angular. The setup of Jasmine could get complicated because users have to select an assertion or mocking library before even programming in it. This third-party dependency was already a red flag to our project due to the need to keep the project lightweight. Jasmine was also missing easy asynchronous testing, which for mocking and running tests that reference a database asynchronously was something that needed to be done.

Jasmine, although big in the industry, was also slowly losing momentum as people are choosing Jest over it. Without these built-in features that Jest offers, Jasmine falls short of a good option.
Support for Jasmine exists, the API documentation was decent and there was a lot of stackoverflow support for it. When dealing with technologies, Jasmine also works- even if it does require another assertion or mocking library. Just like Mocha, having a third-party library for a feature requires more documentation and a more complex setup, even for something as simple as writing unit tests.

Jasmine's syntax was similar to Jest in that every word isn't separated by method chaining; however, Jasmine still relies on method-chaining:

```
expect(add(3, 4)).toBe(7);
```

Method-chaining was extremely prominent in unit testing as it utilizes test-driven and behavioral-driven development. This also makes Jasmine's code very readable. Having built-in assertion methods also makes comparing readability much easier and also makes support much greater as it comes in a nice package similar to Jest with a few less features.

Summary

Jest has the most features that we need along with everything nicely packaged up without any third-party applications, libraries, or other frameworks necessary. A huge plus was already being built-in to React Native which makes set up for Jest extremely easy and minimizes error.

Jest is also a widely utilized framework for testing. According to npm downloads it was way above all other platforms for unit testing. Although this may seem unimportant, using industry standard and widely-used frameworks that are constantly being updated and cleaned are a huge plus:

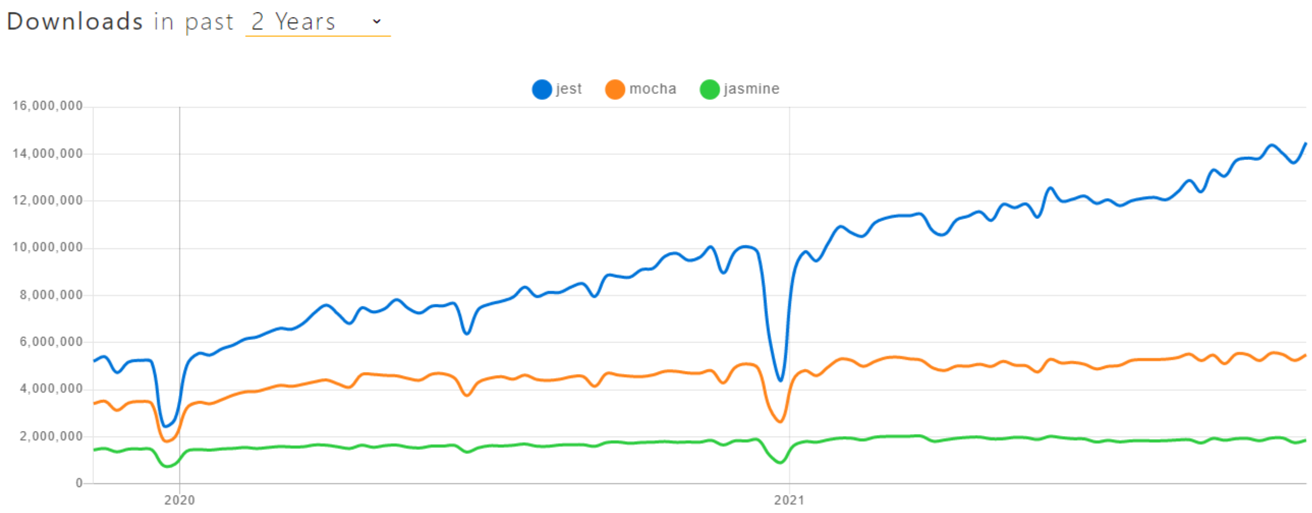


Figure 49 : Downloads

Table 17: Jest

Testing Framework	Ease of Use	Support	Applicable	Readability
Jest	5/5	5/5	Yes	5/5
Mocha	3/5	4/5	Yes	5/5
Jasmine	3/5	3/5	Yes	5/5

10.0 Administrative Content

Information regarding the organization and planning of the team including the creation and discussion of the main project timeline, each group members roll, and the total budget calculation and breakdown are all recorded in this section of the report. While this administrative information did not directly pertain to the design or programming of the final project, the process of gathering information and making informed decisions as a group greatly benefited from the content in this section.

10.1 Milestone Discussion

Below we've created our milestone checklist that would serve as a guide to us throughout senior design I and senior design II in order to keep us on track. For senior design I, the milestones are broken down by what was needed to be accomplished. This would mainly pertain to research, meetings and finalizing the paper. We anticipate having a PCB of our AC/DC converter by the end of the semester. A CAD model and UX design mockup of our device was also a goal we tend to achieve by the end of the semester so that we are able to start senior design II on the right track.

By the start of senior design II, we would have tested all of our equipment needed for our device. During this semester, we would build our final prototype and do an abundance of tests to ensure that every component was working properly. As testing was done, additions to the report would be made. Preparations of our final presentation would occur once the device was fully functional and our paper had gone through its final draft. By this time, each member would have a clear understanding of every component, its functionality and be able to address any questions that may be asked by the faculty.

As shown in the table below, our first tasks were about research and deadlines. As the semester comes to an end, we now have an idea on how our device would operate and are starting to shift from research to testing. In the weeks to come we would've been testing the sensors, microcontrollers, speakers, displays and motors. By doing so we would see what piece of hardware works for our device and where we need to add modifications.

Table 18: Main timeline

Task #	Description	Date Completed
1	Initial project idea brainstorming	8/27/2021
2	Solidifying project idea and organization of group roles	9/3/2021
3	Market research, project design brainstorming	9/10/2021
4	Initial divide and conquer documentation, creation of project specifications, creating of project box diagrams	9/16/2021
5	Project identification meeting, solidifying project design	9/22/2021
6	Divide and conquer adjustments, solidifying programming design	9/30/2021
7	Relevant technology discussion and research, microcontroller research	10/8/2021
8	Safety and standards research, finalizing of constraints	10/15/2021
9	Discussion and research of internal components, acquiring of initial build components	10/22/2021
10	PCB design, 3D CAD model UX Design Mockup	12/4/2021
11	Initial testing of microcontroller and smart phone app communication	11/5/2021
12	Print PCB design, initial printing and assembly of external components UI Design Mockup	01/04/2022
13	Finished first draft of report	12/4/2021
14	Editing and polishing of report	12/6/2021
15	Finalization of report	12/6/2021
	END OF FIRST SEMESTER No additional work was planned to occur during the winter break between semesters beyond the ordering of additional parts and the commission of the PCB board construction	12/9/2021

	BEGINNING OF SECOND SEMESTER	
16	Build prototype	1/20/2022
17	Testing and redesign	03/25/2022
18	Finalize project	04/05/2022
19	Finalize report	04/06/2022
20	Prep for senior design showcase	04/17/2022
21	Present senior design	04/20/2022

Apart from our main timeline, we also input another table to specify what each person on the team was mainly responsible for. It would focus on research and design on the specific part, as well as when it was complete.

Liam Kenney acts as our program manager. He was incharge of setting deadlines and keeping everyone on track. Aside from doing so, he was also the lead when it comes to the implementation of motors. Research and development of the motor was one of his main duties. It was one of the main components of our device as it was what keeps the Pill^3 moving when asked by the microcontroller. He would also be designing the body of our device with assistance from Oriana and Fernando.

Oriana Acala was the lead for the implementation of sensors and microcontrollers. As the lead for sensors and microcontrollers, she would be incharge of researching and testing which type of hardware are needed, where we could obtain it and compare costs of each item. Having a background in software, she would also be assisting Jordan when he needs assistance in the development of the web application.

Fernando Oriundo was the lead of the power supply, speaker and displays of our device. He would be in charge of researching different types of power supplies, speakers and displays, comparing costs and testing to see what best fits the Pill^3. Development of the PCB of the AC/DC converter would be based on his research with assistance from Oriana and Liam.

Jordan Schneider was our lead software engineer. He would be incharge of the app development, and web application. Deciding on the framework, platform and testing of the software was his main responsibility. He would be implementing ideas discussed as a team into the microcontroller. He would get assistance from Liam, Fernando and Oriana when it comes to the overall design and final implementation of the applications.

Table 19 : Research and design

Senior Design 2			
Task #	**Description**	**Lead**	**Date Completed**
1	Frame Design	Liam Oriana Fernando	11/20/2021
2	Microcontrollers	Oriana	11/14/2021
3	Motors	Liam	11/9/2021
4	Sensors	Oriana	11/13/2021
5	Power Supply	Fernando Liam Oriana	11/10/2021
6	Displays	Fernando	11/11/2021
7	Speaker	Fernando	11/12/2021
8	Chassis	Everyone	11/3/2021
9	Web Application	Jordan	3/20/2022
10	App development	Jordan	11/23/2021
11	Components	Oriana Fernando Liam	11/30/2021
12	PCB layout	Liam Oriana Fernando	12/6/2021

The table shown below would discuss how we would build the body of our device. Each person would be responsible for getting tools, equipment and to test out the amount of stress/strain the body could withstand. Since this purely involves only mechanical systems, we decided to separate it from the hardware and software aspect of our design

Table 20: Physical body of device

Task #	Description	Lead	Date Complete
1	Obtain wood and power tools	Everyone	01/05/2022
2	Measure hardware to fit inside the chassis	Liam Fernando	01/10/2022
3	Complete wooden chassis	Oriana Jordan	01/20/2022
5	Redesign sorting disks to be 3D printed	Liam	02/14/2022
7	Compare results	Liam Oriana	02/25/2022
8	Research other alternatives before final decision	Jordan Fernando	03/02/2022
9	Repolish any errors	Liam Oriana	03/10/2022
10	Customize chassis to fit app design	Jordan Fernando	04/02/2022
11	Run final test	Everyone	04/05/2022

10.2 Budget and Finance Discussion

The budget for this project would be self-funded. We have a range of values that could potentially make our product either low cost or very expensive. It would really depend on the components and where we purchase them. Regardless of it , the product would not exceed the $200 mark, this budget would be equally divided among the members of the group. At this point of our design, we have not purchased any major parts, only small components have been attained (e.g., resistors, capacitors, wires). All other main parts would be ordered once the design of the product was finalized.

The idea was to build a product that was not too expensive for the public. Most of these components are affordable. The highest expense might be the main components such as the custom PCB, the Arduino, or any kind of microcontroller we decide on utilizing. Below we could find a cost estimate for each of the components that are predicted for our project as well as the total cost of it. Like it was mentioned previously these are approximations based on current market values, there was a possibility that most of these components would either increase in value or if we are lucky could potentially be cheaper.

As for the design of the Pill^3 chassis, it could be low on cost since the materials could be found in most stores. The total cost of the project falls in a very wide range since it would really depend on what materials we use and the quantity. The table provided was more a reference of the main things we might use for the project; the project alone should have a total cost of around $300 as it was previously discussed in the specifications.

Table 21: Budget

Quantity	Part	Manufacture	Manufacture Code	Price	Description
1	Arduino Nano IoT 33	Arduino	ABX00027	$18.40	The Arduino UNO WiFi Rev.2 was the easiest point of entry to basic IoT with the standard form factor of the UNO family. Whether you are looking at building a sensor network connected to your office or home router, or if you want to create a BLE device sending data to a cell phone,
3	IR Sensor	Esooho	EK1254x5	$8.28	Infrared Sensor could be used for 3-5V DC power supply modules. It has a red power indicator. IR Infrared Obstacle Avoidance Sensor Module could be widely used in robot obstacle avoidance, obstacle avoidance car, line count, and black and white line tracking and so on
3	Stepper Motor	Adafruit	XYU42STH34-0354A	$56.01	A stepper motor to satisfy all your robotics needs! This 4-wire bipolar stepper has 1.8° per step for smooth motion and a nice holding torque. The motor was specified to have a max current of 350mA so that it could be driven easily with an Adafruit motor shield for Arduino and a wall adapter or lead-acid battery. ($18.67 cost per motor)
1	Speaker	Arduino	TPX00080	$7.99	"Stereo Enclosed Speaker" was a new passive speaker with JST PH2.0 interface. It was a perfect option for any DIY audio project which requires an 8 ohm impedance and 3W power.
1	Display	MakerFocus	SSD1306	$7.99	"128x32 OLED display, no need for backlight, self-illumination, Display Color: White. The display performance was better than the traditional LCD display, also lower consumption; SSD 1306 oled display; I2C oled display, IIC (I2C communications) simplifies connections;"

1	Temperature Sensor	Texas Instrument	LMT85LP	$1.60	The LMT85 was a precision CMOS temperature sensor with ±0.4°C typical accuracy (±2.7°C maximum) and a linear analog output voltage that was inversely proportional to temperature. The 1.8-Vsupply voltage operation, 5.4-µA quiescent current, and 0.7-ms power-on time enable effective power-cycling architectures to minimize power consumption for battery-powered applications such as drones and sensor nodes.
1	Power Supply	ALITOVE	CJ-1230	$9.99	"Input: 100V-240V 50/60Hz; output: 12V DC, 3A max; Works with device that draws less than 3A, such as 500mA, 1A, 1.5A, 2A, 2.1A, 2.3A, 2.6A, 3A.DC plug connector external diameter was 5.5mm, internal diameter was 2.1~2.5mm. Comes with a female DC barrel connector. You could easily screw wires on it without soldering."
1	DC Power Jack	DIKAVS	B074LK7G86	$10.99	1.0mm pin terminals perfect for solderless breadboards and PCB boards Accepts 2.1mm x 5.5mm plugs Terminals are also solderable Lifetime: 5000 cycles of insertion and extraction
			Total	$124.25	

A separate table was made for the chassis of our device. We decided to break it down into types of materials that could possibly be used to house our equipment. We did this for the design and research phase of our device. We would take into account many different types of materials. Once we started to test the durability, cost and over appeal, we would be able to finalize our decision. Below we would have broken our chassis into three separate tables since it was made of three separate materials. We would be taking a look at a wooden, 3D printer and acrylic chassis for the body of our device. After much consideration, at our final design, we decided to use plywood as it was the most affordable and durable material that fit our needs.

Table 22: Wooden Chassis

Material	Quantity	Pricing
Plywood 4ft x 8ft	4 pieces	$85
Paint	1 can	$4 - $100
Power Tools	N/A	N/A
Screws/ Bolts	1 box	$8

Total	6 pieces of material	$97 - $193

Table 23: 3D Printed Chassis

Material	Quantity	Pricing
Filament	1	$20 - $40
3D Printer	0 -1	$0 - $300
Total	1-2 pieces of material	$20 - $340

Table 24: Acrylic Chassis

Material	Quantity	Pricing
Acrylic	1 case	$40 - $100
Screws/ Bolts	1 box	$8
Laser cutting	N/A	$0
Total	2 pieces of material	$48 - $140

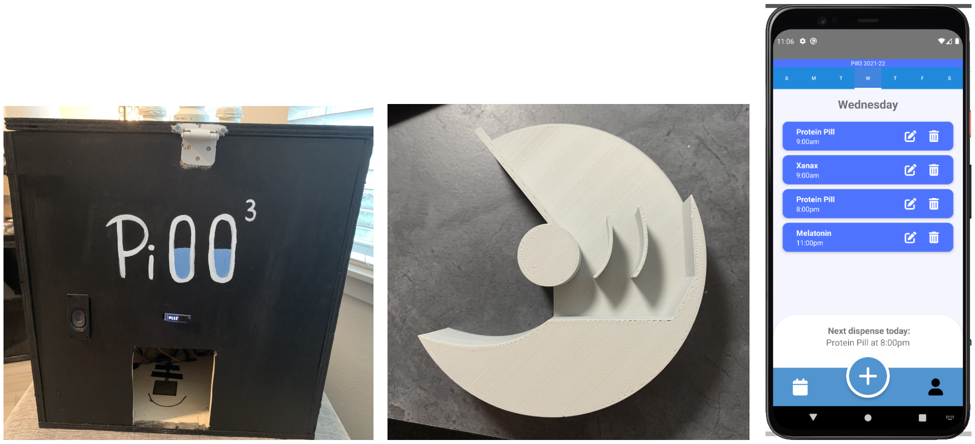
11.0 Project Summary

Below we can see the Pill^3 at its final stage alongside the mobile application. As shown, the final chassis of the Pill^3 was constructed out of six 1.5 ft of plywood. This was to create plenty of room for the storage capabilities of the Pill^3 as well as to ensure that no excess heat generated from the electronics could drastically change the ambient temperature of the pill storage cylinders. We decided that we would also paint the device to have a more consumer friendly and aesthetically pleasing look. The color scheme and design were chosen to match the simplistic blue layout of our mobile application. Hinges were added to the top and back panel of the Pill^3 to allow for ease of access when testing and describing the mechanisms of the device to judges. These hinges will not be in the final product and are only used in this prototype model of the Pill^3. As such, they were not factored into the final calculations for the price of one unit, and the extra screws that would be used to securely seal the box were included instead.

Our final sorting disk was made from filament that we 3D printed, rather than the initially stated laser cut acrylic sheets. The change in material was due mainly to time and cost constraints, as any testing or changes that had to be made to the design of the sorting disk would be almost 100 times more expensive if we kept the original material. Each 3D sorting disk cost approximately $0.20, in contrast to the nearly $20 price tag for each acrylic sorting disk. As for time, the total turnaround time for a single 3D printed sorting disk was close to 10 hours. This estimate included the time it took to send the file to the printer as well as the time to go and receive the printed piece, as the machine used was from a private residence. Meanwhile, the turnaround on an acrylic sorting disk would have been significantly longer, as the only available machine to laser cut the layers was on campus, and wait times to use that machine were unpredictable at best and nonexistent at worst. We would choose to incorporate three of these sorting disks onto the final version of our device, one for each container.

The final version of our mobile application would be connected to our cloud database, Firestore, and would send all information regarding pills to be stored there. The app would then handle all notifications locally, meaning that the connection to Firestore only had to account for one-way traffic. On the other end of the pipeline, the Firestore database would receive REST API commands from the MCU via the Arduino Firebase ESP Client library, which would ask for and then receive information regarding all of the currently scheduled pills. The MCU would do this every 5 minutes, making a total of 288 calls each day to the database, to update the information stored on the hardware as often as possible. The MCU would then check the current time against the desired dispensing time of each pill every 30 seconds, ensuring that each pill would be properly distributed at the time that it was scheduled for.

Together, these components comprised our completed device. The user experience could happen from start to finish, and all requirements that our team declared at the beginning of this project were met. Below are the final versions of the most important components to our functioning device; the exterior of the box, the sorting disk, and the mobile application.



11:06

Pill3 2021-22

S M T W T F S

Wednesday

Protein Pill
9:00am

Xanax
9:00am

Protein Pill
8:00pm

Melatonin
11:00pm

Next dispense today:
Protein Pill at 8:00pm

Figure 50: (a) Chassis of Pill^3, (b) Sorting Disk, (c) Mobile App

11.1 Improvements and Challenges

After two semesters, the Pill^3 proved to be successful as it accomplished every goal that we originally set out to achieve. Throughout this process, we faced many challenges and thought of possible improvements that could be implemented in later versions of this product. Below we have explored several changes that could be made if this project were to carry on, as well as the challenges that our team faced throughout this process of designing and creating the Pill^3.

1. Gel Pills: during the Pill^3 design stage, we initially planned to incorporate all kinds of medication in our device, and as such all pills that we could think of were taken into account. However, there is a huge variety in consistency, size, and shape of available pills. In an attempt to be as inclusive as possible, without overdesign or overcomplicating our sorting methods, we researched the types of conditions and other factors that affect the commercial storage of pills. From this research we reached a few conclusions. Any type of pill that required extremely specialized storage conditions or other extraordinary circumstances to prevent it from going bad could not be used in the device (these pills or other medicine that needed to be in liquid form, or that needed to be kept frozen). An extension of this discovery was that gel pills specifically raised several issues with storage, as they needed to be kept perfectly dry and at exactly room temperature, otherwise they would leak or stick together. As our design had a mostly open storage container for the pills, the natural moisture in the air could cause major issues if a user was to store gel pills. Our temporary solution to the problem was to omit them from use in this maschine, but given more time and research it would be possible to create special atmosphere sealed and temperature controlled containers. This would also broaden our range beyond just gel pills and possibly into more difficult to store medicine.
2. Acrylic sheets: at the early stages of the design we considered acrylic sheets that would be cut to be part of the dispensing system, acting as the rotating and sorting disks. Due to pricing limitations regarding this resource and the significantly lengthier process of prototyping with the acrylic material, we decided to design a 3D model of the disks previously mentioned. This choice alone reduced the final price of the finished product by approximately $40.00 and allowed us more freedom with creating, testing, and adjusting prototypes. If we had stuck with the acrylic sheet, each prototype alone would

have cost upwards of $20.00 and would have had twice the turnaround time that a 3D printed prototype did.

3. PCB iterations: Throughout the Pill^3 design we started with a different microcontroller, which was limited with the amount of digital pins but housed hardware that significantly simplified the digital connection to the database. To solve the limited number of pins available, we added a multiplexer. The idea was to have two of the three stepper motors connected to it, to then connect it to the MCU. While doing so, we encountered a problem where the stepper motors needed constant inputs that we could not replicate using the multiplexer. Due to these circumstances, we decided to re-design the PCB and change the MCU. The second design contained an Arduino nano IoT 33, which provides more analog and digital pins. After testing the design, it showed positive results. To enhance our final design we decided to print one more iteration with some pin configuration changes and mounting holes that previous versions did not have.

11.2 Conclusion

The purpose of the Pill^3 was to provide an affordable and easy to use automated pill dispenser for anyone to use. We hoped that it would help to eliminate the inconsistencies people have when taking their prescribed medication and vitamins, and reduce the amount of injuries or illnesses caused by prescription nonadherence. The easy to use mobile application and low impact housing made it so that anyone would be able to schedule when their medication needed to be dispensed and be notified as to when to take their medication right from their own homes, without the need for medical or technical personnel to set up the device. The visual, audio and smartphone notifications would work in tandem to ensure you would never forget to take your pills when you needed them.

With an overall design consisting of one MCU communicating with six unique hardware components via a PCB and a chassis composed of simple geometric shapes, mass-producing this device would come at a low cost and could easily be integrated into already-present supply lines. We made sure to keep our design simple, yet effective for the purpose we envisioned. Throughout the process, we designed this device with the consumer in mind and as such we were able to achieve all of our goals. Throughout the course of two semesters of senior design, our knowledge as engineers was put to the test. We faced many different obstacles from every facet of product engineering, and overcame them all to create the device we have now. Overall, we were able to implement our gathered engineering knowledge into the creation of this device and achieved this feat as a result of everyone’s contribution. The lessons we learned here will allow us to reach even greater heights.

Bibliography

Connecting Arduino to Firebase to send & receive data. In: Arduino Project Hub. https://create.arduino.cc/projecthub/electropeak/connecting-arduino-to-firebase-to-send-receive-data-cd8805. Accessed 13 Nov 2021

Health-Care was te. In: World Health Organization. https://www.who.int/news-room/fact-sheets/detail/health-care- was te. Accessed 27 Oct 2021

IEEE 1588-2019 - IEEE Standard for a Precision Clock Synchronization Protocol for networked measurement and Control Systems. In: IEEE SA - The IEEE Standards Association - Home. https://standards.ieee.org/standard/1588-2019.html. Accessed 23 Oct 2021

IEEE/IEC 82079-1-2019 - IEEE/IEC international standard for preparation of information for use (instructions for use) of products - part 1: Principles and general requirements. In: IEEE SA - The IEEE Standards Association - Home. https://standards.ieee.org/standard/82079-1-2019.html. Accessed 27 Oct 2021

Magazine S (2014) The environmental disaster that was the gold industry. In: Smithsonian.com. https://www.smithsonianmag.com/science-nature/environmental-disaster-gold-industry-180949762/. Accessed 23 Oct 2021

Mishra A (2015) Academic Journals - Journal of Geography and regional planning - impact of silica mining on environment. In: Journal of Geography and Regional Planning. https://academicjournals.org/journal/JGRP/article-full-text/915EC0C53587. Accessed 23 Oct 2021

MissionCritical, Instructables (2018) OLED I2C display with Arduino Tutorial. In: Instructables. https://www.instructables.com/OLED-I2C-DISPLAY-WITH-ARDUINO-Tutorial/. Accessed 1 Dec 2021

NFPA 70E-2018. In: ANSI Webstore. https://webstore.ansi.org/Standards/NFPA-Fire/NFPA70E2018. Accessed 1023 Oct 2021

P11073-10472 - health informatics--Personal Health Device Communication--part 10472: Device specialization--medication monitor. In: IEEE SA - The IEEE Standards Association - Home. https://standards.ieee.org/project/11073-10472.html. Accessed 23 Oct 2021

Utsource123, Instructables (2019) How to make motor controller circuit. In: Instructables. https://www.instructables.com/How-to-Make-Motor-Controller-Circuit/. Accessed 13 Nov 2021