

GNG2101
Final Design Report

Night Call Bell

Submitted by

CallForCare, B12

Justin Cheng, 300122560

Luka David, 300134324

Avaneesh Madaram, 300130329

Chetan Mandur, 300135400

Marcus Bessette, 300135951

December 3, 2020

University of Ottawa

Abstract

At the start of the semester, our group B12 (also known as CallForCare), was tasked with designing a night call bell to be used by Fran, our patient at Computer Wise. Fran wants an insurance device to call for help when she has issues when she decides to sleep. The goal of this project is to create a device for Fran and her staff to be able to communicate with each other by sending a signal when she needs help. Our task is to design and program microcontrollers to connect wirelessly for Fran to send an alert to her staff during the night. We will need to design a CAD model to contain all the electronic parts and 3D print them.

Table of Contents

Introduction	6
Need Identification and Product Specification Process	7
Conceptual Designs	11
Feasibility Study	20
Project Plan, Execution, Tracking & Bill of Materials	22
Analysis	26
Prototyping, Testing and Customer Validation.	27
7.1 Prototype 1	27
7.2 Prototype 2	34
Final Solution	41
8.1 Hardware	41
8.2 Software	42
8.3 Electronic Components	43
8.4 Testing	44
Conclusions and Recommendations for Future Work	45

List of Figures

Figure 1: CallToU receiver and transmitter

Figure 2: Medical Guardian base system and wrist alert

Figure 3: This is a diagram of the design concept

Figure 4: This is a diagram of *Breadboard & Raspberry Pi Wiring (Untested)*

Figure 5: This is a diagram of the bell prototype

Figure 6: This is a diagram of the coding process

Figure 7: This is the circuit diagram to light up a RGB led

Figure 8: This is the circuit diagram for the button

Figure 9: This is a diagram of the bell and receiver

Figure 10: This is the front of the receiver

Figure 11: This is the inside of receiver

Figure 12: This is the inside side view of the receiver

Figure 13: This is the back side of receiver

Figure 14: This is the Inside of the bell

Figure 15: This is the back view of bell

Figure 16: This is a photo of testing the led

Figure 17: This is a photo of the back of the bell

Figure 18: This is a photo of the front of the bell

Figure 19: This is a photo of the back of the receiver

Figure 20: This is the front of the receiver

Figure 21: Bell Circuitry

Figure 22: Receiver Circuitry

Figure 23: Raspberry Pi Zero Pin Labels

List of Tables

Table 1: This is a table of design criteria and importance

Table 2: This is a table of a list of metrics with associated units

Table 3: This is a table of benchmarking of similar products

Table 4: This is a table of a set of target specifications

Table 5: This is a table of sketches and explanation of design concepts

Table 6: This is a table of the scoring of design concepts

Table 7: This is a table of distributed tasks

Table 8: This is a table of the schedule

Table 9: This is a table of the bill of materials

Table 10: This is a table of data collected for CAD designing

Table 11: This is a table of the testing done for prototype 1

List of Acronyms

1 Introduction

In the GNG2101 course, our group, B12 (CallForCare), was tasked with designing a night call bell for our client, Fran at Computer Wise. Fran is a senior citizen at a retirement home that has trouble controlling her hand/arm movements. Sometimes during the night, she would need help from nurses but she is unable to call for them. After multiple meetings with Fran and having a better grasp of her situation, we deduced that there needed a technical solution for elderly or disabled people to use their voice to signal for help during the night. It functions with the goal to relieve clients with anxieties and worries when being alone and possibly save lives. CallForCare's objective is to provide a reliable night call bell that is activated by voice recognition to send a signal to a receiver that the nurse will carry.

2 Need Identification and Product Specification Process

Problem Statement: Mainly during the night, the client needs a technical solution to use her voice to signal for help to her nurses.

Need #'s	Category	Need	Design Criteria	Importance
1	Primary Needs	Low cost (\$100 budget)	- School provided software/materials	2
2		Easy to use	- Simple shutoff and activation methods	3
3		Reliable	- Not applicable	5
4		User-friendly	- Bell is easy to activate and shut off without any prior experience required to use the bell	3
5		Fast transmission	- Local network	4
6		Easy to spot visibly	- Mounted to a wall with led or glow in the dark material	3
7		Physically shut off	- Manual shut off button	4
8	Secondary Needs	Controllable volume	- Adjustable volume controls	2
9		Voice-activated	- Activated with keywords such as "help" or "hey"	5
10		Wired powered	- Wired powered to the wall plug	4
11		Timer indication to try again	- Lights to indicate how long to wait until alerting the system again	4
12		Lights and sound notification	- Sound and light indication of when help is coming - Sound and light to alert staff	3
13		Portable	- Clip or chain for staff to wear	4
14		Lightweight	- Lightweight material of receiver	2

15		Reliable power source for receiver	- Battery or charging	4
----	--	------------------------------------	-----------------------	---

Table 1: This is a table of design criteria and importance

Metric #	Need #'s	Metric	Importance (1-5, with 5 being of most importance)	Units
1	1	Cost to produce transmitter and receiver	2	\$
2	5	Range transmission of a transmitter to receiver	3	m
3	13	Bell Size	1	cm
4	13	Size of alerting device	3	cm
5	6, 11	Brightness of bell when activated	4	lm
6	12	Noise level that the speaker picks up the voice	5	dB
7	12	Noise level that the speaker lets off to notify the staff	5	dB
8	10	Power source of the bell	1	V
9	14	Weight of the receiver	3	g
10	15	Power source of receiver	4	V

Table 2: This is a table of a list of metrics with associated units

Metric #	Need #'s	Metric	Importance (1-5, with 5 being of most importance)	Units	Medical Guardian (Figure 2)	CallToU (Figure 1)
1	1	Cost to produce transmitter and receiver	2	\$	29.95 (per month)	43.54 (one time)
2	5	Range transmission of a transmitter to	3	m	396	150

		receiver				
3	13	Bell Size	1	cm	15.24x15.24x 7	8.125x8.125x2.16
4	13	Size of alerting device	3	cm	N/A	6x6x2
5	6, 11	Omits light when activated	4	lm	Yes	No
6	12	Noise level that the speaker picks up the voice	5	db	N/A	N/A
7	12	Noise level that the speaker lets off to notify the staff	5	db	60	0-110 dB
8	10	Power Source of the bell	1	V	Continuous/32-hour backup battery pack	Batteries
9	14	Weight of the receiver	3	g	N/A	357 g
10	15	Power source of receiver	4	V	N/A	Batteries

Table 3: This is a table of benchmarking of similar products



Figure 1: CallToU receiver and transmitter



Figure 2: Medical Guardian base system and wrist alert

	Metric	Units	Marginal Val	Ideal Val
1	Cost to produce transmitter and receiver	\$	<100	<50
2	Range transmission of a transmitter to receiver	m	>150	>100
3	Bell Size	cm	<15x15x8	<8x8x6
4	Size of alerting device	cm	<8x8x4	6x6x2
5	Omits light when activated	lm	Yes	Yes
6	Noise level that the speaker picks up the voice	db	>60	>40
7	Noise level that the speaker lets off to notify the staff	db	>60	>70
8	Power Source of bell	V	Batteries or continuous	continuous
9	Weight of the receiver	g	<425	<142
10	Power source of receiver	V	Batteries or continuous	Batteries

Table 4: This is a table of a set of target specifications

Reasoning for Target Specification Selection:

1. Cost is a relevant target specification as the client does not have an endless amount of money to spend on the project. We need to be aware of our limitations so we don't spend too much on the technology we want to use. We can save money by creating and adapting the technology ourselves.
2. The range is a very important specification as without a stable range the bell will not be able to contact a possible receiver. Too big of a range and we are wasting power and resources, and too small of a range will simply not work. This is one of the main specifications we will need to focus on when creating this project.
3. The size of the bell is another target specification as it will limit the amount of technology we can use. As of course, the smaller and more efficient the technology the more it costs. We also do not have enough space in the client's environment to fit an

entire computer to work as a bell. The bell must be able to fit on a nightside table or shelf. Therefore creating this target specification.

4. The alerting device needs to be large enough to hold all the components within it. It needs to be large enough so it can't be easily misplaced/lost. Must also be easily pocketable for staff.
5. A form of activation confirmation must be given to the user. Since the product is primarily meant to be used at night, the lights would be easily visible. By using lights instead of some form of audible confirmation, we are reducing the noise pollution within the house, making it easier for roommates to be kept undisturbed
6. It is unknown how loud our client can project their voice. By targeting a low dB range, we are making it easier for the client to use the system. Also, it is important to target a low dB range as the product is being used at night; we want to reduce the amount of noise pollution created by our client as our goal is to keep roommates undisturbed
7. The noise level that the speaker lets out to notify the staff must be determined as it should not interrupt the roommates in the house. However, it still needs to be loud enough to alert the staff as well.
8. The client specified that they wanted the bell to be powered through the wall outlet as it would be more reliable and more convenient to use as they wouldn't need to replace batteries or charge it. This is for safety measures to ensure that the bell is useful for emergencies at any time.
9. The weight of the receiver should be lightweight and unnoticeable to not affect or restrain the staff at any moment.
10. Since the receiver cannot be plugged into a wall outlet and be portable on staff, it will have to be either battery-powered or chargeable. We decided to go for battery-powered as staff would not be always reminded to charge it.

3 Conceptual Designs

Function Decomposition:

Bell:

Plug-in Device → Monitor sounds for keyword → Recognize keyword → Activate light confirmation → Send alert to receiver → Wait for a response from receiver → Deactivate light → Unplug Device

Receiver:

Wait for alert → Receive alert → Activate alarm → Wait for termination → Return alert

Information Flow:

Monitor sounds for keyword → Initiate Call for help → Monitor receiver response

Material Flow:

Material: Patient, Nurse

Recognize keyword → Activate light confirmation → Deactivate light

Energy Flow:

Plug-in Device → Unplug Device

A
V
A
N
E
E
S
H

Concept 1

Sender Hardware



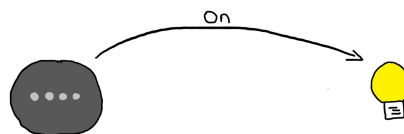
One of the things that the client does during the night is listen to rain sounds and other calming music.

The picture above is a Google Home Mini, a speaker that comes with Google Assistant. Using a Google Home will be helpful as we do not have to worry about these sounds interrupting the user's call for help.

Also using the Google Assistant we can develop a software that can listen for keywords and send a signal to the receiver as soon as possible. With constant updates with Google Assistant, it will be

Concept 2

Receiver Light



This light would be a Smart Light that is turned on by the Google Home. A program can be written that sends an alert to a Smart Light to turn on.


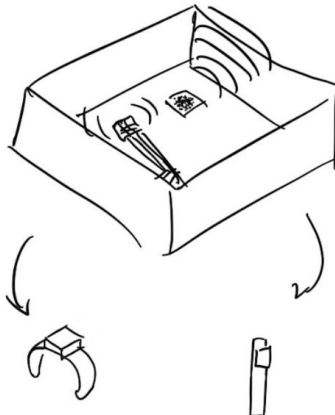
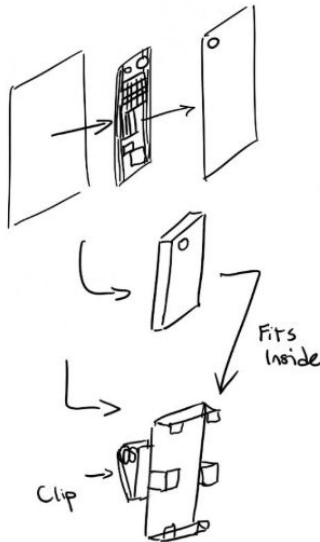
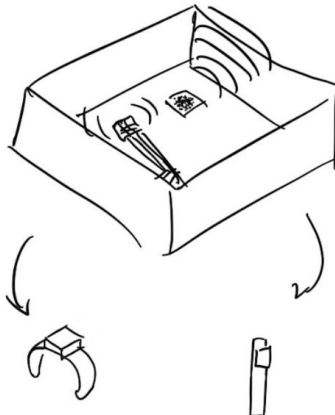
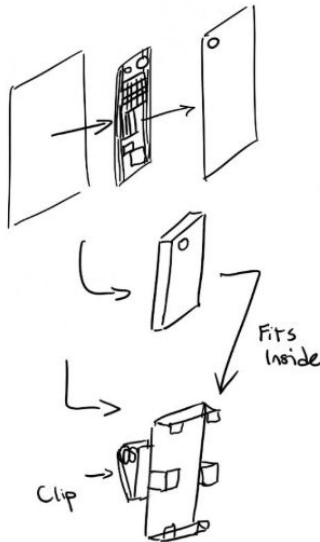
The Smart Light would turn on in the room it is placed. When the light is on it means that the client needs help. The light will then turn off when the receiver goes to the client.

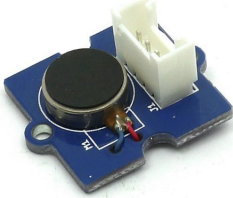

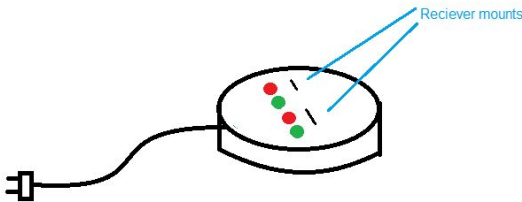
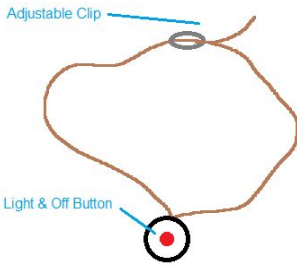

Concept 3

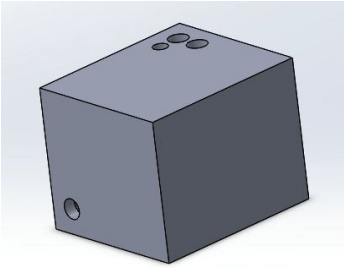
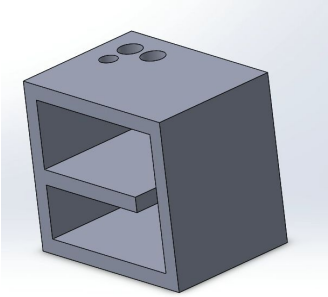
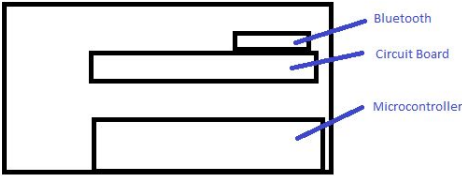

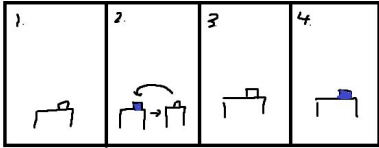
Receiver Design



- A small box that can fit in the receivers pocket
- The receiver would then vibrate whenever it is sent a signal
- The box also has a speaker that outputs the alarm to notify the user that the client needs help
- There is also a red button for the user to click one to turn off the alarm as well.
- This is used to not get in

	easy to keep the device up to date.		the way of the user when they are working with other people in the house
L U K A	<p>Monitor sounds for keyword</p> <p>One of the key functions of our product is to distinguish keywords from random sounds before alerting staff for help. One of the solutions to achieve this function involves speech recognition software. Here is basically how it is supposed to work:</p> <div><div></div><div><p>DIFFERENT SOUNDS</p><p>~~~~~ - constant sound - rain - thunder music for sleeping</p><p>uu uu uu - small noise - creaks</p><p>(uu) (uu) - audible sound</p><p>~~~~~ - loud noise ~~~~~ - yelling</p></div></div> <td><p>Activate alarm (Receiver)</p><div></div><p>One of the functions of the product is to alert the nurse when the bell is activated. To do so I have added some concepts of how we can alert the nurse. To make the most cost efficient product we should create our own hardware and connect it to some smart board. Some possible ways to alert the nurse are through light, vibrations and noise. Light might not be visible in case it is in someone's pocket so the best two technologies to use are to add a small speaker as well as a small vibration motor. These can be easily implemented with a raspberry pi or arduino.</p></td> <td><p>Receiver Design</p><div></div><p>This concept is a design for how the receiver can look and be incorporated into the nurse's daily routine. Basically the receiver will be almost like a remote that can be accessed wirelessly from the bell box.</p><p>The receiver design will be sleek and will have a button or speaker on it if needed. The receiver can be either stored in someone's pocket or attached at the hip by a clip on mechanism. That way it does not have to be on a wrist or around someone's neck. This can be someone from the responsibility to keep their hands away from water.</p></td>	<p>Activate alarm (Receiver)</p> <div></div> <p>One of the functions of the product is to alert the nurse when the bell is activated. To do so I have added some concepts of how we can alert the nurse. To make the most cost efficient product we should create our own hardware and connect it to some smart board. Some possible ways to alert the nurse are through light, vibrations and noise. Light might not be visible in case it is in someone's pocket so the best two technologies to use are to add a small speaker as well as a small vibration motor. These can be easily implemented with a raspberry pi or arduino.</p>	<p>Receiver Design</p> <div></div> <p>This concept is a design for how the receiver can look and be incorporated into the nurse's daily routine. Basically the receiver will be almost like a remote that can be accessed wirelessly from the bell box.</p> <p>The receiver design will be sleek and will have a button or speaker on it if needed. The receiver can be either stored in someone's pocket or attached at the hip by a clip on mechanism. That way it does not have to be on a wrist or around someone's neck. This can be someone from the responsibility to keep their hands away from water.</p>

	<p>Analysis</p> <p>[[[]</p> <p>Sound: ("Hey")</p> <p>"Huh - Ayy"</p> <p>Not similar to:</p> <p>~~~~~ "rain"</p> <p>u u- u "creaks"</p> <p>~~~~~ "normal conversation"</p> <p>((u ((u "thunder"</p> <p>Simon is an open source software that can be implemented for advanced speech recognition. It is very flexible and we can customize it to suit the needs of our technology. It has even more features than necessary as it is not bound by dialect or language. We can create language and acoustic models from scratch which can be useful for creating specialized tests for our individual client. It also has an easy to use interface and we have access to various test cases to improve our product's functionality.</p>	<p>Here is what an actual vibration motor that is capable of connecting to a raspberry Pi would look like:</p>  <p>Also here is a speaker that can connect to a raspberry pi as well.</p> 	
M A R C U S	<p>Receiver Charger</p>  <p>If a mobile receiver is decided upon it will require a battery as a power supply. To charge this battery there are three main options; a wireless station, a single cord for each receiver, or a mounted station. The latter is shown above and is beneficial in that</p>	<p>Receiver Design</p>  <p>A possible design for a mobile receiver could be a necklace that is worn around the neck. The strap</p>	<p>Sound Monitor</p>  <p>A possible design for monitoring key words in the room would be using a speech recognition</p>

	<p>it allows the maximum number of receivers (two in this case) to be charged at once in. Also the coding would not be as difficult as a wireless charging station and also allows leds to be implemented to demonstrate when charging is complete.</p>	<p>would be adjustable that way it would be a one size fits all for the nurses. Simultaneously, this would allow it to be tightened enough to be worn around the risk. The method that informs the nurse that help is required would be a flashing light on both sides of a water bottle cap sized object. To deactivate and send information back that they are on their way the light would also act a button that would be able to be pushed down.</p>	<p>module. The benefit of these chips is that they are very small and consume low amounts of voltage (3.5 to 5 volts). Simultaneously, these recognition modules are cheap meaning the budget can be lowered or moved towards other aspects such as the receiver. Finally, these chips are tiny so the sender does not have to be too large.</p>
	<p>Sender Design +Plug-in Device/ Unplug</p>  <p>Figure 1: Image of the sender device</p>  <p>Figure 2: Image of the sender device cut in half.</p>  <p>Figure 3: Image of the inside the device</p> <p>The concept of the sender device is to have the Arduino and breadboard in it. The breadboard will be on the top section on the</p>	<p>Send alert to the receiver Connect the microcontroller to the local network to execute code and transfer information from one device to another. This plan involves having the Bluetooth component part of the microcontroller or an extended component inside the whole system (Product). The positioning of the Bluetooth device will be on top of the breadboard in figure 3</p> <p>Arduino</p> <ol style="list-style-type: none">1. HC05 Module (Bluetooth Module)2. Ethernet connection <p>Raspberry Pi:</p> <ol style="list-style-type: none">1. Built-in Bluetooth for version 3,4 or Zero W  <p>Figure 3: Image of HC05 Module</p>	<p>Wait for a response from the receiver (LED concept idea)</p> <p>Idea 1: LED</p> <p>The sender device can either use RGB LED or RGB LED strips as indicators. The light activation method is shown below.</p> <ol style="list-style-type: none">1. Default: No lights on2. After calling for help: Pulsing light3. No response: No lights on4. Gets a response: light on fully<ol style="list-style-type: none">a. Need to close manually  <p>Figure 4: Image of Light activation method</p> <p>Idea 2: Speaker</p> <p>Design idea: The speaker or buzzer will be inside the device, but near a hole or opening to easily output audio to the client. This</p>


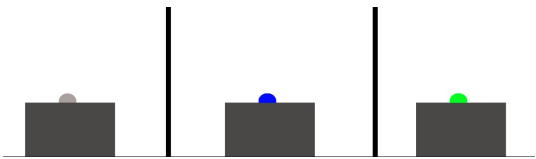
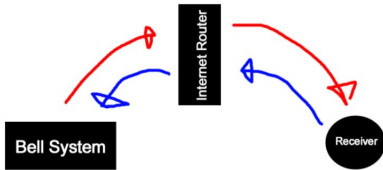

<p>device on top of the Arduino board since the components will stick out after plugging it in on the breadboard. There will be a gap inside where the wires can easily connect onto the breadboard. (See figure 2). On the top, there will be three holes, button, led, and microphone. The button is for when the staff comes to manually shut off the device, the led is to indicate the status of the situation, and the microphone is to pick up her voice. On the side, the hole is for the power plug for the Arduino board which will be connected to the wall outlet using a DC power plug. In figure 2, the level inside the device is not necessary if we are able to attach the breadboard to the side (See figure 3). This concept design can also be used for a Raspberry pi microcontroller but the dimensions will be different.</p>	<p>We can possibly use a Serial Voice Recognition module</p>  <p>Figure 5: Image of Serial Voice Recognition module</p>	<p>design concept is to solve any problems if the client has issues or trouble with seeing the LED's on the device. The Arduino can output audio of specific words or sentences to indicate if the signal was sent or not. The buzzer idea can be used as a siren to indicate an emergency or that the client is waiting for a response from the receiver.</p> <p>Arduino:</p> <ol style="list-style-type: none">1. Arduino speaker module power amplifier2. Piezo Buzzer
<p>Light Activation</p>  <p>(Left image)</p> <ul style="list-style-type: none">- Device doesn't detect key word- Light is off- Signals that no alarm has been sent <p>(Middle Image)</p> <ul style="list-style-type: none">- Device detects keyword- Light turns blue- Signals that alarm is ringing, not been acknowledged by receiver (caretaker) <p>(Right Image)</p> <ul style="list-style-type: none">- Receiver (caretaker) acknowledges alarm- Light turns green- Signals that caretaker is on their way	<p>Long Range Communication</p>  <ul style="list-style-type: none">- Use LAN to send information between bell and receiver- By using the local network system, we have a large range of connection to work with<ul style="list-style-type: none">- A standard 2.4Ghz router have a range from 140-300ft- Less interference, no need to add a new constant signal through the air- Well documented- As their internet infrastructure gets better, so does the connectivity of our product- Con: internet must always be on for system to work	<p>Receiver Design</p>  <ul style="list-style-type: none">- Wearable on the wrist- Adjustable Strap- Big red button to accept alerts- Green LED that flashes when alert is going off (that hasn't been accepted)- Speaker to enable small beeping alert- Square shape (better housing of parts)- Enough room for small computer boards (eg. Raspberry Pi Zero)

Table 5: This is a table of sketches and explanation of design concepts

Product Concept	Scoring	Reason
Marcus 1	3	Not cost efficient; adds extra unnecessary coding; allows for multiple charges at once and the leds give a visual to when they are fully charged; not as portable as say a phone charger
Marcus 2	4	Most optimal location as it doesn't get in the way of the nurses; size of necklace can be tightened; light on both side allows it to be visible at all times; and if noise is used its closer to the ear of the nurse so the decibel level can be lower
Marcus 3	2	Low voltage and small size are optimal; however unfamiliar with the board type unlike a raspberry pi or arduino board;
Justin 1	5	This design is intuitive, cost efficient and easy to produce. The simplicity of the design looks sleek and will protect the hardware inside the product. It will also have space for the functional components inside the device. Although we might not need a circuit board and microcontroller the layout makes a lot of sense.
Justin 2	4	The products that have been researched are applicable to the client's issues and can be implemented easily. The raspberry pi is a great selection as it is simple enough for us to work with yet can be complicated as much as we need. We can add various technologies to it to have all the functionality we need.
Justin 3	4	This product concept has some very applicable ideas with the light verification, and some decent ideas with the sound. The pulsing light would work well as it would easily distinguish between when the alarm has been received or not. Also they can be easily implemented.
Avaneesh 1	1	Google Home will be hard to work with; Requires hard coded keyword to even activate (unable to keep activated); Must work through Google API, unable to sideload our program
Avaneesh 2	2	As stated above the Google Home design would not be practical. In return means that the design won't work but smart lights are still a good idea. By using smart lights, we are able to alert the caretakers without adding to the noise pollution of the house. The only issue is to find smart lights with an open API so that we can integrate them with our software. Smart lights are also very expensive, so it acts more of a luxury feature.
Avaneesh 3	4	The design is simple to create; able to house all the components; and intuitive to use. The only issue with this device is portability; if the user does not have enough pocket space (or any) to carry the device, they might find it clunky to carry around.
Luka 1	4	By using an open source voice recognition software, we are able to focus more of our time on creating the hardware and programming scripts needed for the product to work. If we were to make our own

		voice recognition software, it would have been a long process and with the amount of time we have, we wouldn't have been able to create an accurate system (voice recognition needs a lot of testing and data feeding).
Luka 2	4	Compact enough that it doesn't need to be altered depending on if we use a watch or necklace; it has a vibration motor and a speaker which also allows for two different methods to alert the nurses; compatible with both a raspberry pi as well as an arduino board.
Luka 3	3	This would only work if we were to use a design similar to Avaneesh Concept 3. This would elevate the issue of the user not having space in their pocket to hold the receiver. This design would allow the receiver to be out of the user's way. Only issue is that it relies on the user to have somewhere comfortable to clip the receiver.
Chetan 1	4	By using rgb lights instead of blinking/flashing it allows for easier understanding by the client of what is taking place in terms of if their voice was picked up; brightness is an issue in that it must be visible at all times and wherever the client is in their room.
Chetan 2	3	This product concept is applicable as using wifi is a great way to link various devices in one household. The one issue however is that we need to constantly have the devices connected to the wifi, and that may not be the case if the wifi is turned off during the night or shuts down on occasion. Would not be a reliable design.
Chetan 3	3	Wrist watch is beneficial in that it is almost always visible which means a noise wouldn't be necessary and only a light or vibration would be needed; however for nurses the watch might get in the way when doing things such as washing dishes and helping patients.

Table 6: This is a table of the scoring of design concepts

Chosen design

For our design we have chosen to use elements of Justin's sender device to create our own technology within a small box. We will use a Raspberry Pi Zero W as the microcontroller using its built-in bluetooth as per Justin's receiver concept design. The receiver will be in the form of a necklace or a clip on device depending on the weight as in Luka's receiver design. The receiver will either have a breadboard or PCB to connect all the components together. To alert the staff, the receiver will have a built-in speaker, LED, and a button to ping the sender. The sender will use Chetan's specific lighting scheme by using an RGB LED to alert the client. A microphone will be connected to the sender to pick up the client's voice to call for the staff. We will power the sender using a micro USB charger adapter to connect to the wall outlet and have the receiver be battery powered.

Design reasoning

As a group we decided it would be easier and more reliable to use a microcontroller instead of trying to override Google Home Mini. We decided to use a Raspberry Pi Zero W instead of an Arduino microcontroller since the Raspberry Pi Zero W has a built-in bluetooth connector to reduce cost fees as we would need a microcontroller and bluetooth device in receiver and sender. The receiver is preferably better as a necklace or clip on to avoid getting the device wet or damaged. We decided that the receiver should have both a speaker and LED to fully alert the staff. We did not think that the sender needs a speaker as the RGB LED will illuminate the room. Fundamentally, we choose design concepts that best suits our target specifications to fulfil the clients needs.

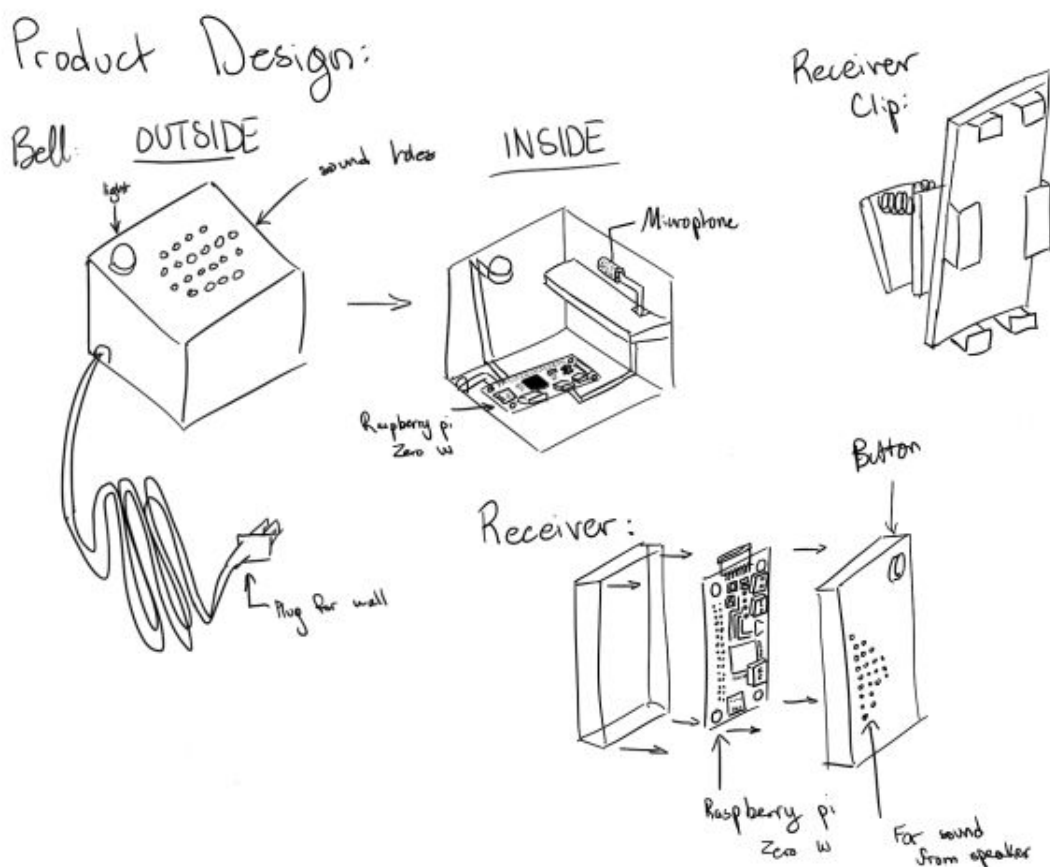


Figure 3: This is a diagram of the design concept

Concept relationship to target specifications as benefits and drawbacks

The target specifications were a good guideline to determine design concepts that would fulfil all of the clients needs, but limited our creativity of the project. The target specification ensures that our product is going in the right direction by having to face limitations and manufacturing problems.

4 Feasibility Study

Identify and discuss the uncertainties and risks associated

I) One uncertainty presented is the decibel levels that the client's sound machine lets off while she is asleep. What we do know is from her is that the sound often used is a constant rain effect with lightning scattered throughout. The problem arises that if the decibel level is too high on the receiver to ignore the lightning but also cannot pick up the client's voice then it would be pointless. But if we put it too low that the lightning causes the device to send signals to the receiver then the nurses would always be tending to the client.

II) Another uncertainty with the design of our project is the distance that the nurses will be from the client/sender. If a bluetooth raspberry pi is used to send the signal a class 2 version has a version of 10 meters. So depending on the layout of the building and maximum distance that is possible the class version of the raspberry pi might need to be changed affecting cost.

III) A final uncertainty would be how a power shortage would affect the connectivity between the sender and the receiver. For example, if one night the power goes out in the house and then comes back on momentarily after, would the bluetooth connection between the sender and the receiver automatically search for each other or would a manual connection have to be done.

Conduct a feasibility study and provide a report that discusses each of the five TELOS factors as they relate to your plan and your concept.

Technical:

For this product, we are using both hardware and software. For hardware, we are planning on using a small computer board (Raspberry Pi Zero W) which has a big open-source community on the internet. The Raspberry Pi Zero W also comes with a lot of documentation which helps with understanding the board and creating products with it. Since a lot of our features (voice recognition and LED) require hardware knowledge, it is helpful that we have members in both Mechanical and Electrical Engineering. For software, we are coding programs that will communicate between all the hardware pieces attached to each Raspberry Pi (eg. LEDs) as well as communicating between different Raspberry Pi's. Raspberry Pi works well with Python, and we have three members on the team with previous experience and knowledge with Python. This is helpful as it means we are able to create sufficient code without much stress.

Economic:

For this product, we are not using any expensive parts. The most expensive product is the Raspberry Pi Zero, with the board alone costing \$14(CAD) and ~\$40 with a kit (pin adapter, power brick, etc). Other parts (RGB LEDs, microphone, and speaker) costing <\$40 (CAD) all together. The reason we chose a Raspberry Pi Zero over other Raspberry Pis/computer boards is for the price per performance. While the Raspberry Pi Zero may not be the most powerful board, it is powerful enough to handle the programs we are going to run on them while having enough processing headroom to ensure that the board

isn't overheating all the time. By keeping the parts cooled, this ensures the longevity of the boards/products. If we were to use another computer board, we would have a board that is overkill with performance, as our program would not utilize all the power. This would cause additional cost for little performance gain.

Legal:

The product is being built using copyright free software and legal materials. The software we are using is open source allowing anyone to use it and share with others. There are no issues with privacy as the client needs supervision during the night which will be when the product is being used. Adding on, we are not involving phones with our product as it is a concern for an invasion of privacy.

Operational:

The biggest organizational constraint is that our team is not all located within one another. This presents a problem when trying to create an actual prototype for the project. Not only do we not have access to the machinery and devices at University but we also are unable to collaborate with each other. Meaning that it may force one person to create a majority of the physical project. Being that our project is decently coding heavy we are able to work on that at home and send that to each other. Another organizational constraint is that we are not only working on this project throughout the first semester. This means that our meetings and work periods may be restricted to certain time schedules that allow for the majority of us to be there.

Scheduling:

The schedule for the product is having a prototype ready by the end of each month. This means that during each month the group will build up on different aspects of the product. During each month there are smaller deadlines where the team has to complete a task each week. This ensures that the product will meet the standards the team has set for themselves. The deadlines are reasonable as each member has a week's time to complete a small task. With the completion of small tasks the prototype can be made easily as the previous steps are done. In the first month the team is planning on creating a basic prototype in which the product's basic hardware structure, voice activation software, and LED's is complete. The second month the team will create internal hardware for the products and software for the sender to alert the receiver. In the third month the team will put everything together and do bug testing to make sure the final product performs up to standards. These deadlines are set in order to not overwhelm the team with work.

5 Project Plan, Execution, Tracking & Bill of Materials

Task	Duration	Members
Buying materials	2-7 days	Everyone
Voice activation software	3-4 weeks	Avaneesh, Chetan, Luka
Bluetooth connection	3-4 weeks	Avaneesh, Chetan, Luka
Sending alert software	3-4 weeks	Avaneesh, Chetan, Luka
LED on receiver	1 week	Marcus, Justin
LED Codes	3-4 weeks	Avaneesh, Chetan, Luka
LED on sender	1 week	Marcus, Justin
Speaker	2-4 days	Marcus, Justin
Speaker code	3-4 weeks	Avaneesh, Chetan, Luka
Sender casing	4-7 days	Marcus, Justin
Receiver casing	4-7 days	Marcus, Justin
Assembly of sender	1 week	Chetan, Marcus, Luka, Justin
Assembly of receiver	1 week	Justin, Avaneesh, Marcus
Debugging session 1	2-4 days	Avaneesh, Chetan, Luka
Debugging session 2	2-4 days	Avaneesh, Chetan, Luka
Debugging session 3	2-4 days	Avaneesh, Chetan, Luka

Table 7: This is a table of distributed tasks

Date	Task
Deliverable D: Prototype 1 (10/8/2020)	Debugging session 1 Buying materials Voice activation software LED Codes
Deliverable G: Prototype 2 (11/5/2020)	Debugging session 2 Voice activation Speaker code Speaker LED on receiver LED on sender

	Sending alert software
Deliverable I: Prototype 3 (12/3/2020)	Debugging session 3 Assembly of receiver Assembly of sender Receiver casing Sender casing

Table 8: This is a table of the schedule

Item	Justification	Quantity	Cost (\$)
General Use			
Raspberry Pi Zero W Link	Microcontroller for the receiver	1	US 14.00 (Gotten from Makerspace)
Raspberry Pi 3 B+ Link	Microcontroller for the bell	1	US 35.00 (Gotten from Makerspace)
Jumper Cables (30 pack) Link	Wires that will be used to connect all electrical circuits	1	US 2.25 (Gotten from Makerspace)
Wall Adapter Power Plug for Pi Zero W Link	This is used for the receiver to have a constant power	1	US 5.95 (Gotten from Makerspace)
Wall Adapter Power Plug for Pi 3 B+ Link	This is used for the sender to have a constant power	1	US 8.00 (Gotten from Makerspace)
Breadboard Link	This is to put all the components on	2	US 9.90 (Gotten from Makerspace)
Microphone Link	This is the microphone for the Bell	1	CAD9.95
LED			
RGB Led Link	LED is used to signal light activation and to get the colour pink	2	US 2.05 (Gotten from Makerspace)
100 Ohm 5% Resistor Link (6 needed for circuit)	Resistors are used to reduce the current	1	US 1.2 (Gotten from Makerspace)
Vibration			

Vibration Motor Link	Vibration device to alert staff on the receiver	1	US 2.15
Transistor NPN BC337 Link	Transistor is used to make the vibration motor work by amplifying or switching the electronic signals and electrical power to it.	1	US 0.50
Diode Rectifier - 1A 50V Link	Used as a semiconductor	1	US 0.15 (Gotten from Makerspace)
1K Ohm 5% Resistor Link	Resistors are used to reduce the current	1	US 0.95 (Gotten from Makerspace)
Button			
Tactile Button Link	The button is used for sending a signal from the receiver to the sender.	1	US 0.5 (Gotten from Makerspace)
330 Ohms 5% Resistor (Could also use 220 Ohms) (Need 2 for circuit) Link	Resistors are used to reduce the current	1	US 0.95 (Gotten from Makerspace)
TOTAL : CAD 118.53			
TOTAL of products purchased: CAD 13.4			
Battery			
Battery pack Link (for pi 2)	This is to power the mobile receiver	1	US 39.99

Table 9: This is a table of the bill of materials

Justification for choice of microcontroller:

The Raspberry Pi Zero W is a cheap option as a microcontroller with a built-in Bluetooth device. If we were to use an Arduino board, it would be larger in size and the external Bluetooth device is not built-in, which would have cost more. It has also a lot of online resources as it is a very popular project resource and has many customizable parts to it.

Justification for choice of battery pack:

For the battery pack, we have selected to use a prebuilt battery controller and battery pack developed by PiSugar that can easily connect to the receiver. This may be the most complicated part of our project as it is very difficult and expensive to power a portable receiver to vibrate or send an alarm on command. The biggest challenges with this design are that the battery has to be large enough to power the circuit for 10-14 hours during the night, as well as optimally not require charging every day, or throughout the night. Being able to turn off the receiver would also be beneficial as we could save power over the day instead of having it be constantly on. Saying that, having a large battery and creating the hardware to monitor the battery requires a lot of experience. The majority of our group has never worked with this type of hardware before or has very little experience.

Due to those reasons we will be buying a prebuilt battery controller that will be able to connect the battery to the raspberry pi as well as monitor the battery life. PiSugar battery components also allow the user to turn off and on the device and recharge easily.

6 Analysis

No calculations were needed for this project

Part	Dimensions
Raspberry Pi Zero W	65 x 30 x 5 mm
Raspberry Pi 3 B+	90 x 60 x 22 mm
RGB Led	Diameter: 5mm
Button	Diameter: 3.5mm

Table 10: This is a table of data collected for CAD designing

7 Prototyping, Testing and Customer Validation.

7.1 Prototype 1

Prototype Objective

For prototype 1, the main objective is to finalize our design concept and have all group members on the same page. We will start researching and learning how to complete subsystems in our system.

Physical Prototypes

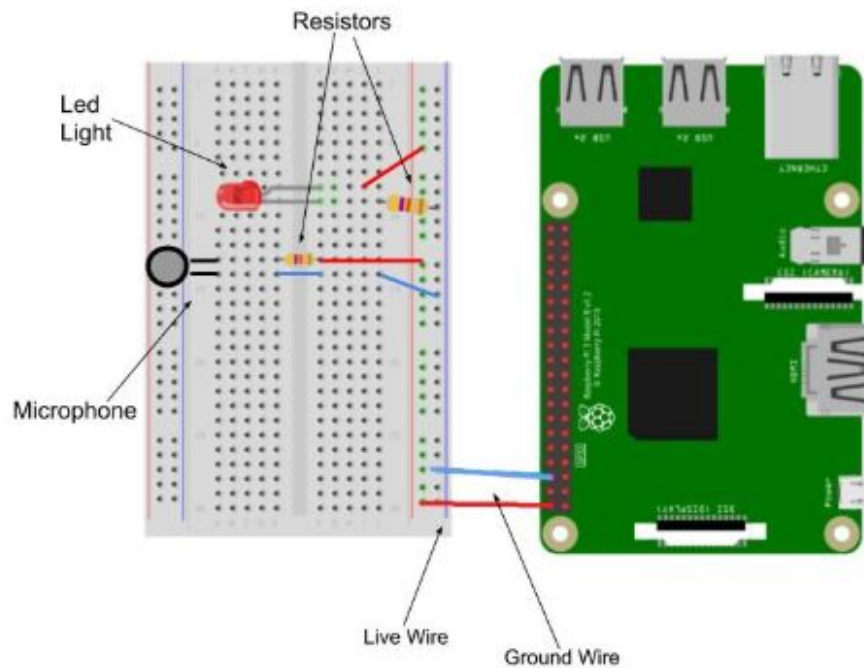


Figure 4: This is a diagram of *Breadboard & Raspberry Pi Wiring (Untested)*

The above image demonstrates an untested version of a raspberry pi (on the left) and a breadboard (on the right) wiring to power the caller the client will use. The blue wire acts as a ground for the arduino board while the red is the live power wire. Resistors are connected to stop the led light and microphone from pulling to much power and frying either or of them. The exact value of the ohms needed to prevent this are not determined as we are yet to select which LED light and microphone we are using.

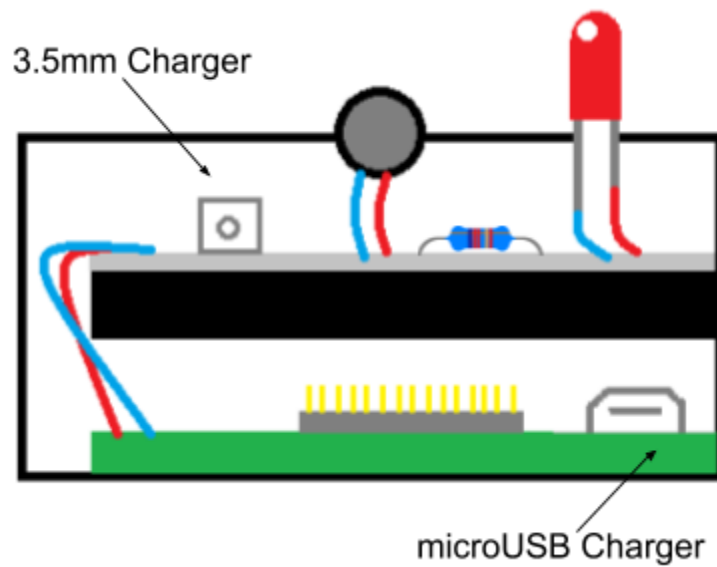


Figure 5: This is a diagram of the bell prototype

In terms of the caller prototype we decided most likely to use a 3D printed case that contains two separate levels. The bottom level would have the raspberry pi and a lot for a microUSB so it can be plugged in at all times. A notch on the left would allow wires to run up to the breadboard in which the light and speaker will be put through slits in the top. This allows for light to be seen better by the client as well as the microphone picking up her key words easier. If a microUSB charger is not decided upon we can also install an external 3.5mm port into the arduino board and use that instead.

Software Prototypes

The software will be fully embedded in the hardware which means that there will be no User Interface.

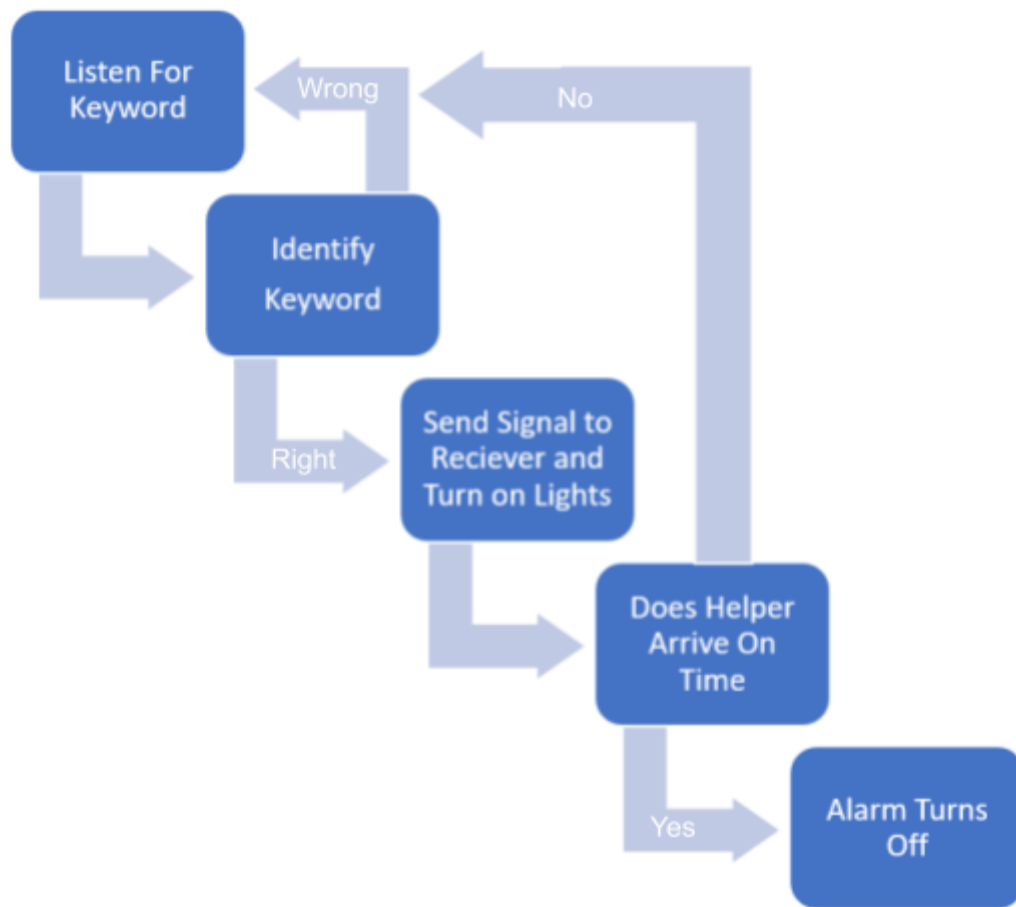


Figure 6: This is a diagram of the coding process

Listen For KeyWord

- The Bell is on during the night waiting for the user to say the key word

Identify KeyWord

- The Bell will identify words throughout the night
- If the word is identified it sets the alarm

Send Signal to Receiver and Turn on Lights

- Receiver starts its alarm
- The lights near the bell turn on

Does the Helper Arrive on Time

- If the helper does not arrive on time then the user will have to activate the Bell again
- If the helper does arrive on time the helper can turn off the alarm using the receiver

Set up RGB LED

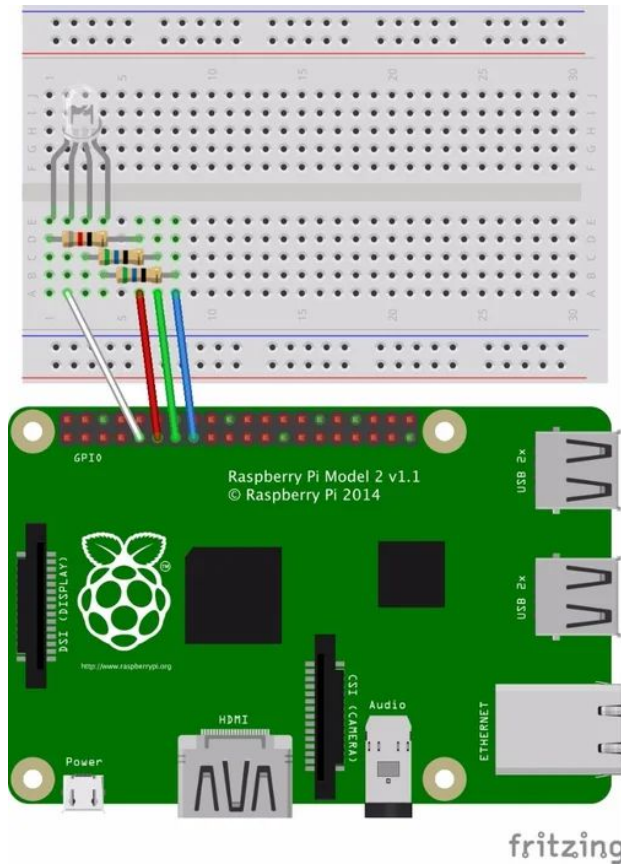


Figure 7: This is the circuit diagram to light up a RGB led

First we have to set up the RGB LED's on the breadboard (Rest is Software). Then we use Python to set up the LED. Using Python allows us to create multiple effects with the lights which are needed for the Call Bell. In Python we will create multiple functions that will turn on/off the light, cause the light to turn pink, and create a pulse effect on the lights.

```
def magentaOn():
    blink(redPin)
    blink(bluePin)

def magentaOff():
    turnOff(redPin)
    turnOff(bluePin)
```

Setup for button:

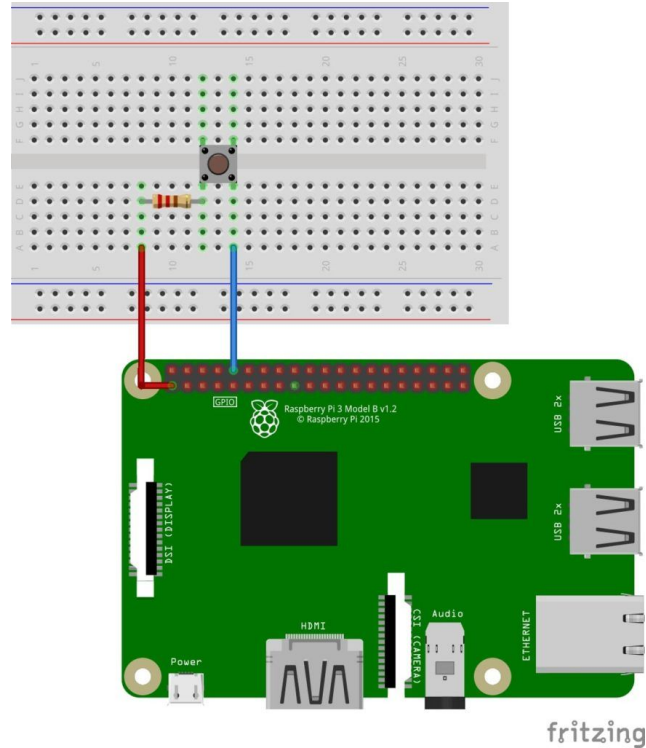


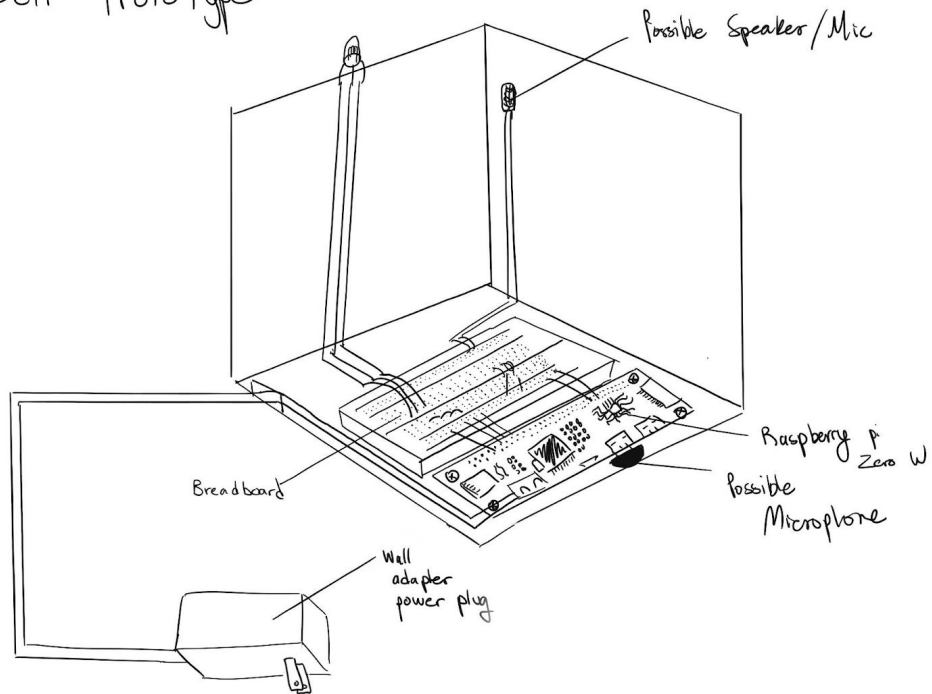
Figure 8: This is the circuit diagram for the button

After setting up the button on the breadboard we will use Python to access the functions. In Python we will create functions for the button. An example of this is creating a function that turns off the RGB's when the button on the receiver is pressed

Product Prototype Diagram:

We have included an updated version of the product prototype, including our decisions and updates. This new prototype includes the necessary resistors, cables and breadboard functions to connect to the raspberry pi. Most of the designs will follow our original design however, we now know what cables to use and where, as well as what devices to use.

Bell Prototype



Receiver Prototype

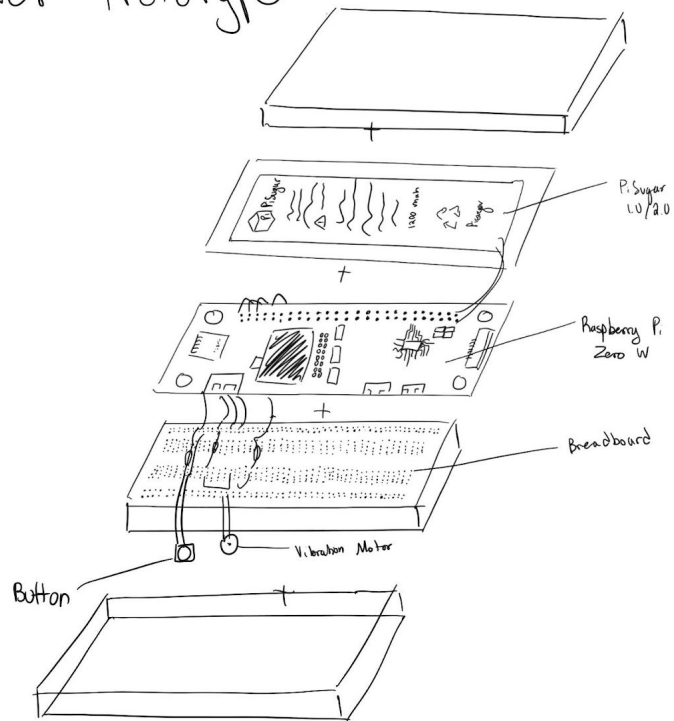


Figure 9: This is a diagram of the bell and receiver

Testing

Since our first prototype will be similar to our final product we are researching and planning out the hardware in great detail, and thus making us unable to write and test code, this testing phase will consist of us testing our ideas by reading through documentation.

Test name	What we did	Findings	Target Spec Dependency
Communication between two Raspberry Pi	Looked through product documentation alongside open-source projects/libraries	It is very possible to communicate between two Raspberry Pi Zeros through both Wi-Fi and Bluetooth. For Bluetooth, we found an open-source library called BlueDot which allows for one Pi to be the receiver and the other to be the controller. Since it is open-source, we can make use of all its functions and mold it how we please	Range transmission of a transmitter to receiver
Powering the receiver	Looked at spec sheets provided by the Raspberry Pi foundation, alongside implementations from the community in similar projects	It is possible to power the Raspberry Pi Zero through a portable battery. The Pi Zero, a 3.7V power source is needed. There are premade battery + controller components that are made for the Pi Zero such as the PiSugar 2 . By using a premade component similar to this, it would allow us to save a lot of time while providing us with a multitude of feature (eg. battery status)	Power source of receiver
Bread Board & Raspberry Pi Connectivity (Caller)	We used a third-party website that tested the wiring of our breadboard	The wiring as shown in the first prototype image seems to be functional. However, we took this information with caution since the simulation tested was between a breadboard and arduino and as of this prototype we are using a breadboard and raspberry pi.	Bell Size & Size of alerting device

Table 11: This is a table of the testing done for prototype 1

Customer Feedback

- The client thinks the idea is great, liked the idea for the clip-on and manual shutoff button
- The client wants the receiver to vibrate, enough for the person wearing it to notice
- The client will provide an audio file that is similar to a real-life simulation where Fran will be asked to do it at night. She will also provide the audio of the rainmaker with or without Fran's voice in the background
- The device should not be flashing for the sender, pulsing is fine
- The colour preference is pink (Fran's favourite colour) and we should avoid blue. The colour pink helps her remain calm
- The client specified that the light mechanism should be simple. She suggested that when she pressed the button the light turns pink and when the staff receives the signal and responses back to the sender, the light will turn off. If the receiver does not send a signal back in a certain amount of time, the sender will start to pulse.

7.2 Prototype 2

Prototype Objective:

Our objective was to develop software elements for our bell and receiver to use since we were not able to start assembling or purchasing parts. We planned to create the code after researching the receiver led and button code as well as the voice recognition code. We also planned out the design of the bell through CAD and tested the circuitry of the LED.

Physical Prototype:

CAD Design:

Receiver CAD Model

Dimensions: 80mm x 47mm x 126.5mm (Walls 6mm on all sides)

The design concept is to have the circuits on the breadboard and stick out on the front surface of the receiver and bell shown in Figure 1. The plan is to create holes/sockets for the components to stick out when the circuit wiring is finished. The front surface is slidable shown in Figure 3 where the lid will be thicker to not slide off and support the front panel.

The green part in Figure 2 is the model of a PiSugar2 with the exact dimensions. There is a socket in the model shown in Figure 4 to charge the PiSugar2.

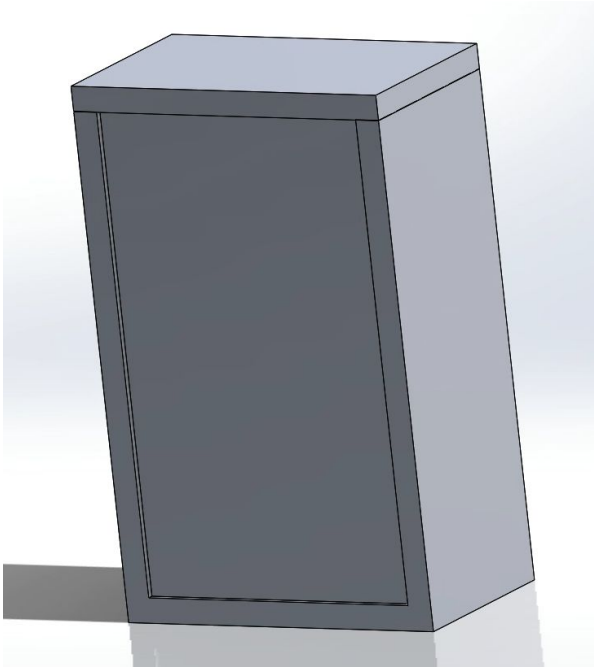


Figure 10: This is the front of the receiver

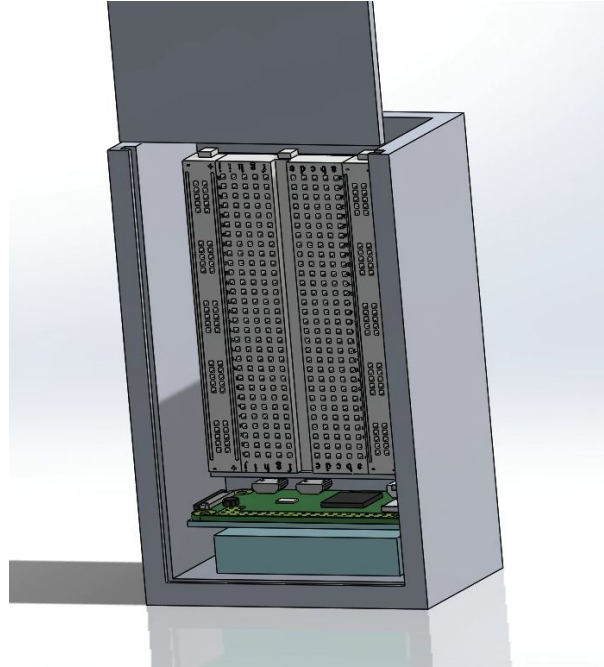


Figure 11: This is the inside of receiver

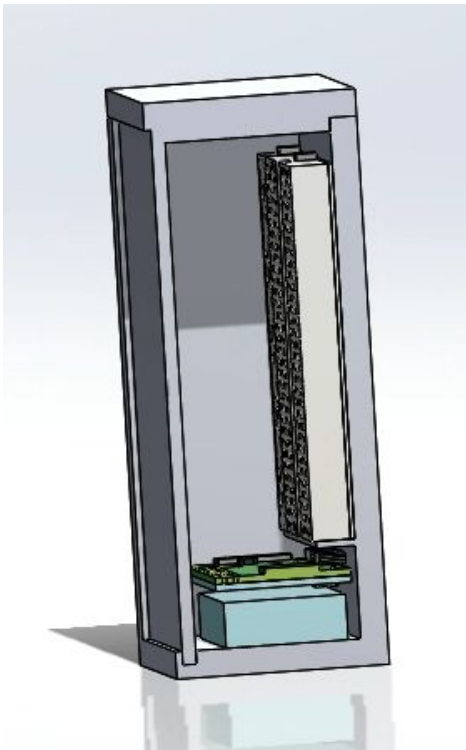


Figure 12: This is the inside side view of the receiver

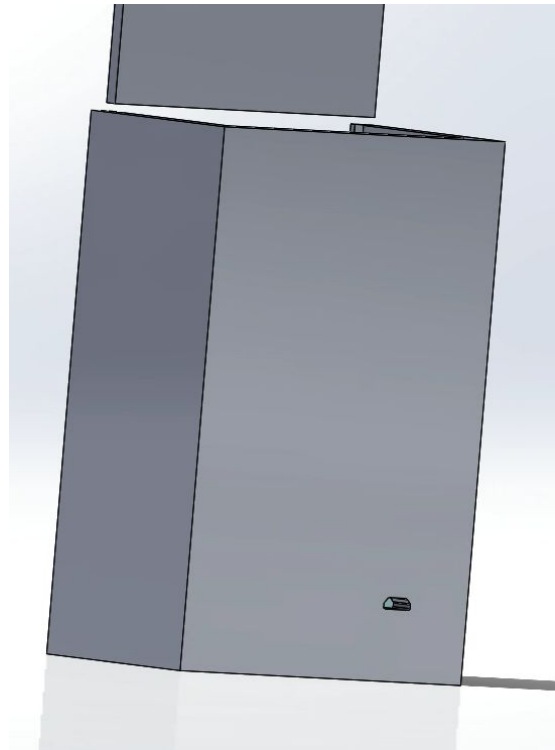


Figure 13: This is the back side of receiver

Bell CAD Model

Dimensions: 80mm x 47mm x 112mm (Walls 6mm on all sides)

The bell has the same design as the receiver, but it is shorter in height as there is no need for the PiSugar2 shown in Figure 5. At the back, there is a port to charge the Raspberry Pi Zero shown in Figure 6.

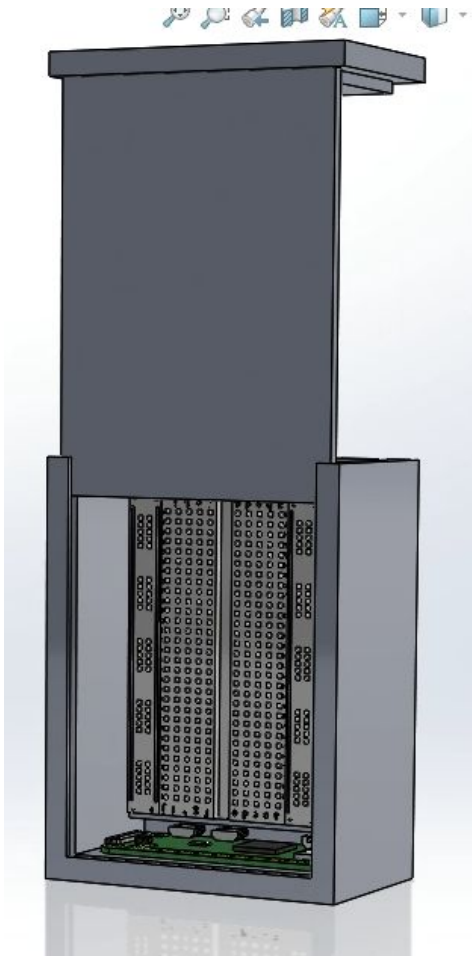


Figure 14: This is the Inside of the bell

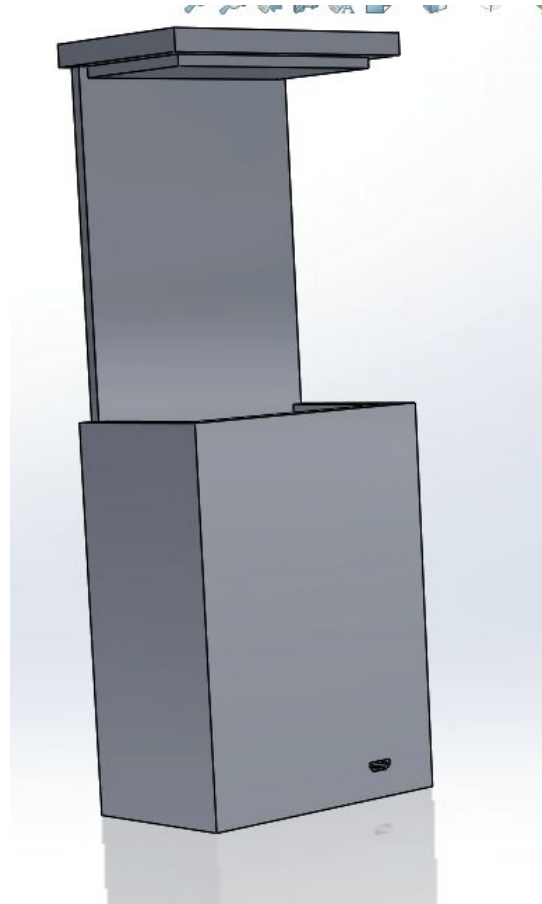


Figure 15: This is the back view of bell

Software Prototype

Currently the work has consisted of attempting to code or develop software for the bell and receiver. The code we have completed is based heavily upon the Bluedot API which allows for simple communication between two raspberry pi's. Raspberry Pi's come with a library that allows programs to be written while utilizing the GPIO pins on the Pi. This is important because it allows us to have a direct connection to hardware components (eg LEDS) through code. Also it is important as it allows us to have a direct connection to hardware components (eg. LEDs) through code. This library is very developed, has

strong support with common hardware components (again, such as LEDs). While we can develop code, this code is essentially ‘pseudocode’ as we have no way of testing it yet. This is due to the fact that we currently do not have any components to work with. This means that we are not able to test out anything that requires the GPIO pins/hardware. We also cannot properly test the Bluedot API and the communication between two Raspberry Pis. All the code could very easily not work as it is based upon many assumptions on critical aspects of design. Also we have been researching to find the right voice recognition software to use. Currently we have found the following software programs to use for voice recognition.

- <https://jasperproject.github.io/>
- <https://www.bishopph.org/step-by-step-raspberry-pi-offline-voice-recognition-with-sopare/>

However, in order to use this software we need to change the Raspberry Pi Model that we are using. This means that we have to change our Raspberry Pi Zero to another model because it can not run the voice recognition software. The other models we might use are a Raspberry Pi Model B, Raspberry Pi 2, or a Raspberry Pi 3.

Testing

LED Testing

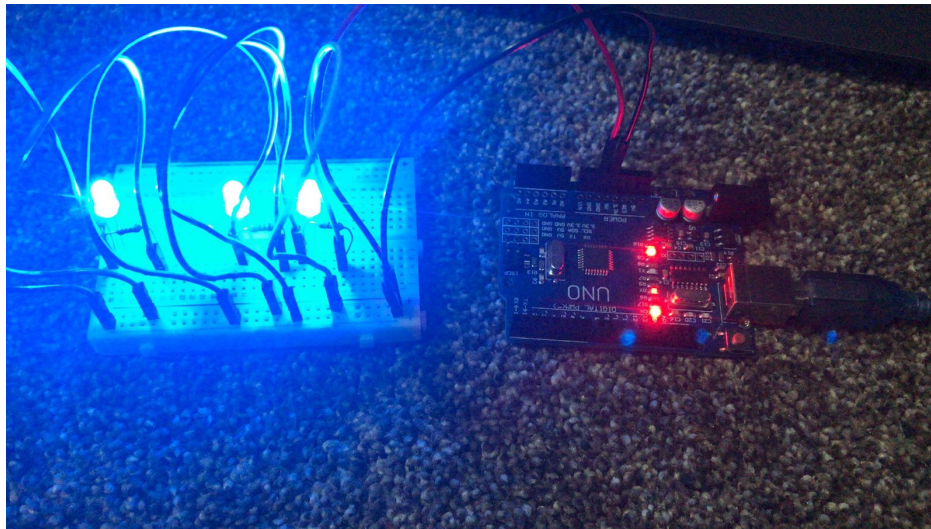


Figure 16: This is a photo of testing the led

For circuitry testing we realized that we could use the same breadboard and circuit layout from laboratory five as long as we decide to use three different coloured leds (red, green and blue). In the case shown above we just used the three blue leds and the arduino board for power since our bill of materials wasn't approved for the other leds and raspberry pi at the point of testing.

However, if we decide to instead use a four prong rgb led we would have to instead have 1 wire for the ground and then three different resistors each connected to their own wire. We are likely to

commit to the second option as it makes assembly easier with only a single led needed to stick out of the casing, less wiring, and a the entire rgb spectrum for lighting.

The expected results for the three led layout was exactly what was given to us in the lab. What we expected four the single four prong led is three a programmable light activates when a certain sound decibel is picked up.

Bell Code

```
//Bell
from bluepy import BlueDot # pip install bluepy
from pynput.mouse import Button, Controller #pip install pynput
import gpiozero

voiceTrigger = voiceRecog
bd = BlueDot()

keyword = [] of keywords
counter = 0

led = gpiozero.LED(17)

def main(): #Main loop
    while (True):

        while ((voiceTrigger.getRecentWord() not in keyword) and counter ==0):
            #keep searching

        keyword_found()
        voiceTrigger.stop()
        click_mouse()
        counter = counter +1

        bdHost.when_pressed = button_pressed

    if counter == 100:
        led.off()
        counter = 0
```

```
def keyword_found():  
    led.on()  
def button_pressed():  
    led.off()  
  
def click_mouse():  
    mouse = Controller()  
    mouse.press(Button.left)  
    mouse.press(Button.left)
```

This will be the main code running at all times on the bell. This code is partly written in pseudocode as there are many aspects of the software that are to be determined (eg. voice recognition software). It is very possible that this code will not work as many of the assumptions are based upon the voice recognition software, which is an integral part of the code. If any assumptions turn out to be not true, it could very well invalidate the whole pseudocode, causing us to redevelop an algorithm.

The expected results for this code is to: detect a keyword (provided in an array list) using the voice recognition software; send a signal to the receiver that the keyword has been said (using Bluedot API) and turns on LED; begin a counter that counts for 100 ticks (after 100 ticks are passed, LED is turned off).

Receiver Code


```

//Receiver

bd = BlueDot()

def main:
    while true:
        bd.wait_for_press()
        keepGoing = True
        while keepGoing:
            speaker.makeNoise
            if (IRL_button.pressed == true):
                keepGoing = false
            click_mouse()

def click_mouse:
    mouse = Controller()
    mouse.press(Button.left)
    mouse.press(Button.left)

```

This will be the code that will be constantly running on the receiver. This code is mostly complete as all dependencies (libraries) have been located. The only thing left to do with this code is test it's functionality. As we currently do not have any of the Raspberry Pi's, this code is based upon many educated assumptions. These assumptions were made by reading the documentation of the necessary libraries and understanding how they fundamentally work.

The expected results for this code is to: wait until a signal is received from the bell (this signal indicated that the bell has been pressed); once a signal is received it causes the speaker to keep ringing/making noise; when the speaker is making noise and the user clicks the physical button on the receiver, the noise stops and a signal is sent back to the bell (indicating that the alert has been received and acknowledged).

CAD Testing and Design

We tested if all the components are able to fit together into the casing of the bell and receiver. We determined that the dimensions for the receiver should be 80mm x 47mm x 126.5mm and the bell dimensions are 80mm x 47mm x 112mm. The expected result was that the components would fit and if they did not, then we would change the measurements according to the testing.

Customer Feedback

- CAD model is good
- No worry or concerns for model design

8 Final Solution

8.1 Hardware

For the final cad designs which were 3D printed using PLA plastic. There are sockets for a button, led and external slots on the raspberry pi's. The top panel is connected to a front panel that can slide right into a slot to contain all the components. The bill of material can be found in Table 9, which will outline all the materials needed to build the product.

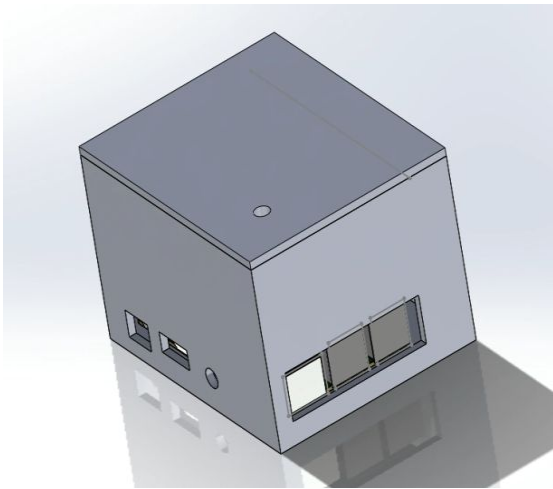


Figure 17: Photo of the back of the bell



Figure 18: Photo of the front of the bell

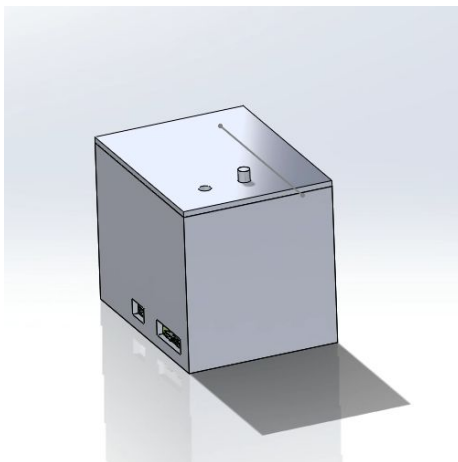


Figure 19: Photo of the back of the receiver

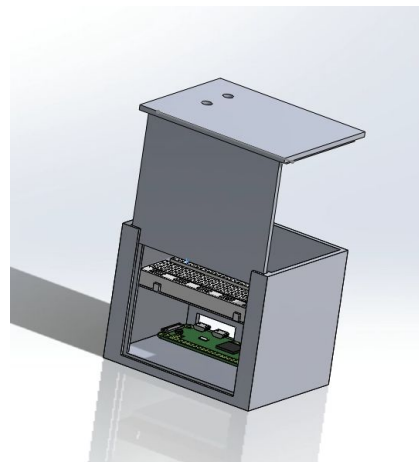


Figure 20: This is the front of the receiver

8.2 Software

All the software that is used is programmed using Python. The voice recognition library we used was from : <https://pypi.org/project/SpeechRecognition/> . Also we used BlueDots library for the Bluetooth connection between the Pi's (<https://bluedot.readthedocs.io/en/latest/>). We also used <https://gpiozero.readthedocs.io/en/stable/> to control the LEDs which would light up when the correct word is identified.

```
recieverCode.py > ...
1  from bluedot.btcomm import BluetoothServer
2  from signal import pause
3  from gpiozero import RGBLED, Button, LED
4  from time import sleep
5
6
7
8  led = LED(2) #Pin layout for the LED
9  button = Button(4) #Pin layout for button
10 waiting = False #Checks if the receiver got sent a signal
11
12 ##Start animation
13 for i in range(2):
14     led.on()
15     sleep(1)
16     led.off()
17     sleep(1)
18
19
20 ##This function will run each time the button is pressed
21 def button_pressed():
22     global waiting
23     if (waiting):
24         print("accepting keyword")
25         led.off()
26         s.send("keyword received")
27         waiting = False
28     else:
29         print("bell did not send an active signal")
30
31
32
33 ##This function will run each time data is received from the server
34 def data_received(data):
35     global waiting
36     if (data == "stop detected"):
37         print("stop detected")
38         waiting = False
39         led.off()
40
41     elif (data == "keyword detected"):
42         print(data)
43         waiting = True
44         led.on()
45
46
47 button.when_pressed = button_pressed #Tells the button what function to run when pressed
48 s = BluetoothServer(data_received) #Initiate the receiver
49
50 ##Main loop that runs forever (or until keyboard interruption)
51 while True:
52     pause()
```

```

1 from bluepy import BlueDot #pip install bluepy
2 from bluepy.btcomm import BluetoothClient # https://bluepy.readthedocs.io/en/latest/btcommapi.html
3 from signal import pause
4 from gpiozero import RGBLED, Button
5 from colorzero import Color
6 from time import sleep
7 import speech_recognition as sr
8
9 r = sr.Recognizer() #Starts the voice recognition
10
11 led = RGBLED(red=9, green=11, blue=10) ##Grabs the pins for the RGB led
12
13 keepGoing = True #Variable that stays true as long as reciever has not been found/connected
14
15 ##Function that runs when data is recieved
16 def data_recieved(data):
17     print(data)
18     led.color = (0,0,0)
19     print("receiver pressed the button")
20
21 ##Start animation
22 for i in range(2):
23     led.color = Color('pink')
24     sleep(1)
25     led.color = (0,0,0)
26     sleep(1)
27
28
29 ##Loops that runs until reciever has been detected and successfully connected
30 while keepGoing:
31     try:
32         c = BluetoothClient("raspberrypi", data_recieved)
33         print("Receiver found!")
34         keepGoing = False
35     except:
36         print("Receiver not found... keep searching")
37
38
39
40
41 ##Main loop for the program, runs forever (or until keyboard interruption)
42 while True:
43     try:
44         with sr.Microphone() as source: #Takes the microphone as input
45             r.adjust_for_ambient_noise(source)
46             # r.energy_threshold = 0
47             data = r.record(source, duration=3)
48             text = r.recognize_google(data, language='en')
49
50             #list of keywords for the bell (both activation and stop keywords)
51             keyWords = {"help", "hey"}
52             stopWords = {"stop"}
53
54             #runs if a stop keyword is detected
55             if any(word in text for word in stopWords):
56                 print("stop detected")
57                 c.send("stop detected")
58                 led.color = (0,0,0)
59
60             #runs if an activation keyword is detected
61             elif any(word in text for word in keyWords):
62                 print("keyword detected")
63                 c.send("keyword detected")
64                 led.color = Color('pink')
65
66
67             print(text)
68
69     except Exception as e:
70         print("Nothing has been detected...")
71         pass
72
73

```

8.3 Electronic Components

In terms of circuit components we used online circuit creators such as tinkerland to create diagrams. From this we replicate them using a breadboard and the two raspberry pi's instead of

8.4 Testing

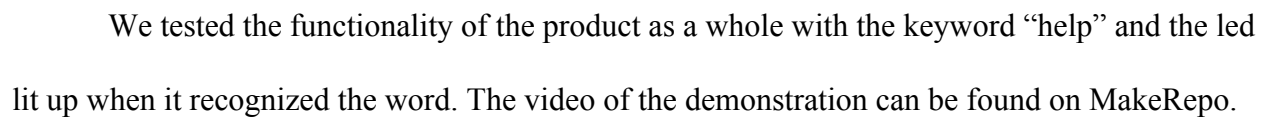


Figure 23: Raspberry Pi Zero Pin Labels

9 Conclusions and Recommendations for Future Work

Throughout our time working on this project, our group faced many challenges, overcame obstacles, and learned a lot of new and very interesting practical skills that we can translate to our careers and future upcoming classes.

We learned that even though there is a plan and even though we want a particular feature on our final design, it may not happen. We had many ideas for the project that didn't get executed, but we simply needed to continue to focus on what was most important, such as the functionality of the product. We also learned how important the presentation of our project was. We were shown how confidence goes a long way and how if we believed in our project, the judges would as well. All the members of our group learned the importance of communication with their teammates so that everyone could stay on the same page and know what was going on at any given time. All of these little things we learned are going to help us be better students and better engineers in the future.

For future work, it is recommended that starting as soon as possible for prototyping and 3D printing will help with producing a better product. A breadboard is not needed if students are proficient with circuitry and soldering.

APPENDICES

APPENDIX I: User Manual

First, the user must plug in both the bell and receiver into a wall outlet. There will be a led that lights up for a brief second to indicate that the device is on. The device will remain on until the power plug is disconnected. Next, plug the microphone into the USB socket in the bell. When the device picks up the keyword, it will light up both devices and only turn off when the button on the receiver is pressed to indicate that help is on its way.

For safety guidelines, keep out of range of children and areas of water or any sort of liquid. If an error occurs when trying to install the device, make sure to check if the microphone and power plugs are connected properly.

APPENDIX II: Design Files

The design files can be found in MakerRepo: <https://makerepo.com/jchen525/b12callforcare>

Files that are included are

- Photos of the cad model
- User manual
- One minute pitch
- Zip file including the cad designs
- Video demonstration