Project Deliverable D: Detailed Design, Prototype I and Bill of Materials

GNG2101 [A03] – Professor Hanan Anis

Submitted by:

Team A2

Ayesha Khan, 300056179

Alessandro Furlano, 300116738

Aunonto Bhuiya, 300115942

Ethan Chan, 300006808

Dieudonne Lomamba, 300095212

Due Date: 08/10/20

University of Ottawa

Table of Contents

List of Figures

List of Tables

1 Introduction

The objective of this document is to present the work done by Team 2 so far on this project, and present the knowledge gained from testing Team 2's work. To divide up their work before the client meeting, the group was split into a software team and a hardware team. Prototypes were developed from both of these groups and were based on the feedback and understanding provided from their first and second client meetings. These were then tested with previous specifications in mind, and then presented to the client along with further inquiries. Afterwards, their decisions on design aspects were finalized allowing them to create a bill of materials for future fabrication. Meeting with the client, and testing the previously proposed ideas, Team 2 has developed a further understanding of what needs to be done for this project.

2. Client Feedback and Critical Assumptions

2.1. Client Meeting Preparation

In order to prepare for their second meeting with Fran and her support staff, Team 2, compiled the following list of questions, their top concept drawing and reviewed their specifications. It was found that their initial meeting had cleared up major areas of confusion in the form of "who", "what", "when" where" and "how". The remaining questions related more so towards the specifics of the device and its feasibility.

1. Is standard phone vibration fine?
2. Can the team get Audio Recordings of Fran? Directly into mic, Distance between bed and side dresser, 1m, 2m.
3. Does a side dresser work, how big is it?
4. Mic placement? Dresser? Or closer?
5. Are there colours that you have difficulty seeing?

2.2. Client Feedback Summary

Upon a pleasant client interview with Fran and her support staff, Team 2 compiled their feedback into **Table 1.** Their feedback was then analyzed to determine what changes would need to be made to the design. The statements were organized into 3 categories: "Improvement", "Neutral" and "Important note". Items in the "Improvement" category were statements that changed the design to better suit the client. Next, the items in the "Neutral" category were statements that provided clarification but did not necessarily change the design. Finally, the "Important note" category were statements that provided good alternatives in the case that the current design fails.

Client Feedback	Improvement
Vibration needs to be strong enough to wake up staff	Improvement on vibration idea as now having three different levels of vibration
Potential use of an alarm (sound notification on portable unit) - staff may be sleep	Improvement - the team hadn't considered that the staff may be sleeping
Option 2 of mic with wire is preferred	Neutral - provided clarification but didn't add to the concept
Wants a dramatic change in colour when signal is sent and when signal has been confirmed	Improvement - the team didn't specify the difference in colour.
No flashing lights, not too bright as well as calming colours is preferred	Improvement - this will help ensure that the device doesn't scare Fran
Vibration can not be strong enough to make the staff uncomfortable and encourage them to remove the portable unit	Improvement - the team didn't consider the staff's mental state.
Has no difficulty distinguishing colours from each other	Neutral - this provided clarification but did not change the design.
Side dresser is fairly big, no size constraints.	Neutral - This provided clarification but did not change the design.
If the main unit is small enough in size, it can be mounted on to the bed, as a solution for the mic.	Important Note - this created a new alternative for the mic option in case an external mic is expensive.
They suggested that the device be triggered based on noise level in the case that there are troubles distinguishing "hey" and "help" from other noise.	Important Note - the team has been given a new fallback option

Table 1: Client Feedback and Improvements

2.3.Critical Assumptions

Based on the client feedback from the team's second meeting with the client and the list of needs and specifications that were determined in Deliverable B the following list of critical assumptions was curated:

1. The API is able to determine if “hey” and “help” have been said.
2. That Fran’s voice will travel to the mic.
3. The limitations for the vibration strength.
4. The strength of the brightness of the large LED.
5. That the devices can be connected to each other over Wi-Fi.
6. That the staff will keep the Portable unit.
7. The 9v battery is powerful enough to run portable unit for a sustainable period of time.
8. That the staff will either charge or replace the battery of the portable unit.

With these assumptions made, Team 2 has thought about several factors to take into account. The first assumption is that the API will be able to pick up when Fran says “hey” or “help”. The team made the assumption that Fran’s voice will be able to travel to the mic. Hopefully, with Fran’s audio recordings the team will be able to determine how sensitive the microphone has to be. Next, the vibrations have to be strong enough to get the attention of the staff worker, but not too strong so that they take it off. The team also made the assumption that the staff worker will keep the portable unit on themselves at all times. Finally, the battery is assumed to be strong enough for the portable unit, so that it can be used for a sustainable amount of time. Depending on if the final design will have a rechargeable battery or just a regular one, the staff will have to either charge or replace the battery. Of the critical assumptions, there were some that, due to their nature, can’t be tested. These assumptions typically to the users after the device would have been fully developed. The remaining assumptions can’t be tested at this stage because of the prototype build that the team decided upon. As described below, Prototype I am more abstract in nature and as such, exact functionality can not be tested.

3 Prototype I

Team 2 decided to have Prototype I as follows:

Prototype type: Focused software, comprehensive hardware

Focused software: Demonstrate speech to text recognition by uploading a recorded voice line and getting the returned text. Pseudocode/flowchart.

Comprehensive Hardware: Design hardware in OnShape, Cardboard (optional)

3.1 Software

For Prototype I, Team 2 decided to write pseudocode for the main and portable units. This code is used to describe the main functions of the device and determine if there are aspects to the design that are lacking, or if all needs have been met.

3.1.1. Main Unit

To begin, the main unit pseudocode has been split into 5 main functions: Server, Audio, Connection, Confirm and Arrived. Please Refer to **Appendix A.1** for the full pseudocode.

Server Function:

The entire code starts in the Server Function. The purpose of this function is to check if a connection has been established to the internet server. The server will be used to connect the main unit to the portable unit, and so, it is integral to the system that it is connected to the internet server. If the main unit successfully connected to the server, the **Connection Function** is then called. Otherwise, the unit will try to connect three times before displaying a red light in the large indicator and flashing a red light in the small indicator.

Audio Function:

Once connected to the server, the main unit will then continuously check to see if there is a noise being registered in the microphone. This is an endless loop of sorts and so, the device will keep checking until a noise is registered. Once a noise is detected, the unit will check to see if the noise is “hey” or “help”. Should the noise be one of those, the **Connection function** will be called in order to send a signal to the portable unit. If not, then the function will just continue to loop.

Connection Function:

The Connection Function is used to check if the main unit is connected to the portable device and then send a signal to it. Once called, the function will try to send a signal to the portable unit, if the signal is sent successfully then the main unit's large indicator will turn yellow in colour and the **Confirm function** will be called. Otherwise, the main unit will attempt to send the signal three more times before turning the large indicator red and flashing a small red indicator and calling the **Server Function** to retest the server connection.

Confirm Function:

The Confirm Function is used to check if the "Ok" button has been pushed on the portable unit. This function continues to loop until a signal is received and once received the large indicator will turn green. This will inform the user that the staff have heard their call and that help is on the way. Additionally, the **Arrived Function** will be called.

Arrived Function:

This function checks to see if the "Here" button has been pushed on the main unit. If the button hasn't been pushed it will continue to loop. On the other hand, if the button is pressed, a "Stop signal" will be sent to the portable unit letting it know that it can now stop vibrating and/or ringing. Finally, the **Server Function** is called thus, restarting the process once again.

3.1.2. Portable Unit

The potable unit pseudocode is used to describe the core functionality of the device, as well as finding out if there are any design ideas that are not up to par. The pseudocode has been split into 4 main functions: connection, server, confirm and neglect. In the connection function it makes sure that the portable unit is connected to the main unit via Wi-Fi and if its not it will turn the led red and have the motor rumble after 3 minutes of disconnect. Next up is the server function, in this the function checks constantly if a signal for help has been sent from the main unit. Next, the confirm function checks if the “ok button” on the device has been pressed, and if it hasn't then led into the next function. The final function is the Neglect function, in here it checks if the button has been pressed as well as if the button on the main unit has been pressed. If both conditions are met, then the vibration and the audio chime will be stopped. If it hasn't been met, then the vibrations will start to increase. Please refer to **Appendix A.2** for the full pseudocode.

3.1.3. API

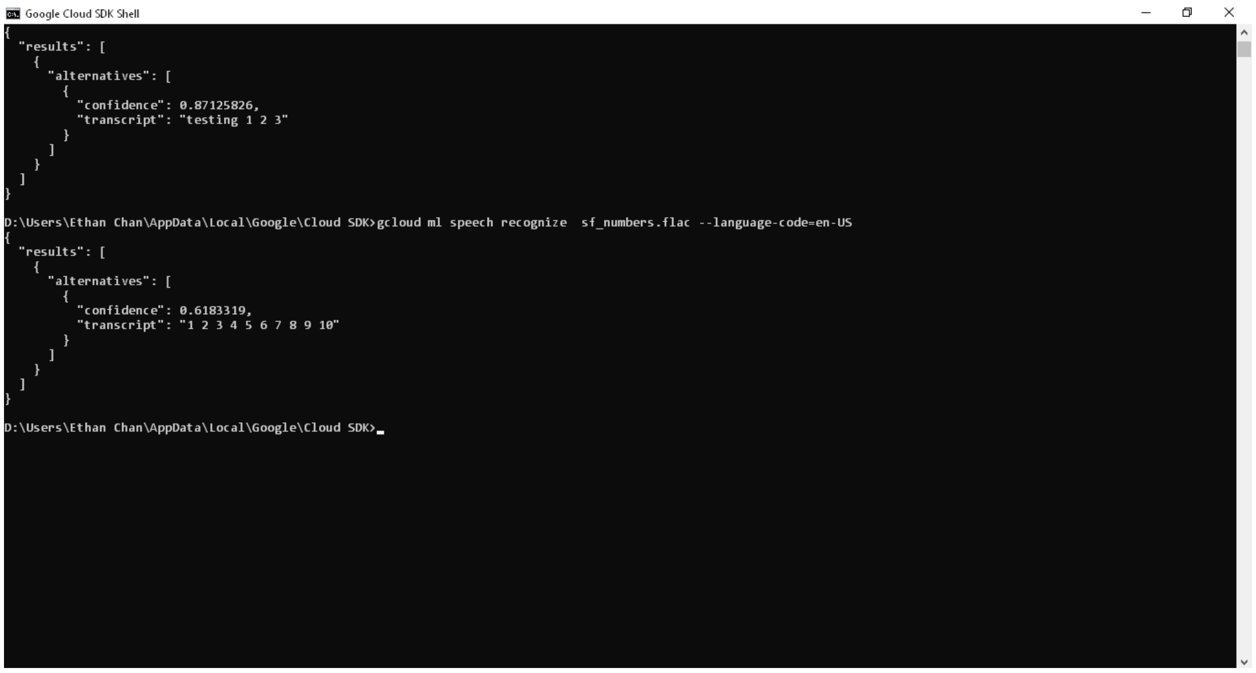
Microsoft Azure:

This API was found to be too expensive for single person usage. The “basic plan” for their speech-to-text API costs over $100 a month for usage. Thus, this was abandoned.

Google speech-to-text:

Google’s version of the recognition software costs significantly less, but still has fees. After the first 60 minutes of usage a month, the client would be charged $0.006 / 15 seconds of audio. While not ideal, The team moved forward with testing it in the case the client wishes to use it. The programming languages that could be used for this include: Protocol, gcloud, C#, Go, Java, Node.js, PHP, Python, Ruby. For this stage, gcloud was used. To transmit a sound file to the API it must first be in the flac format in mono channel mode (These can be changed but are the default settings for it). Then, the line “gcloud ml speech recognize sf_test.flac --language-code=en-US” is used. Returned from this is (confidence being 0 to 1):

```
{
  "results": [
    {
      "alternatives": [
        {
          "confidence": 0.8712586,
          "transcript": "testing 1 2 3"
        }
      ]
    }
  ]
}
```



Google Cloud SDK Shell

```
{
  "results": [
    {
      "alternatives": [
        {
          "confidence": 0.87125826,
          "transcript": "testing 1 2 3"
        }
      ]
    }
  ]
}

D:\Users\Ethan Chan\AppData\Local\Google\Cloud SDK>gcloud ml speech recognize  sf_numbers.flac --language-code=en-US
{
  "results": [
    {
      "alternatives": [
        {
          "confidence": 0.6183319,
          "transcript": "1 2 3 4 5 6 7 8 9 10"
        }
      ]
    }
  ]
}

D:\Users\Ethan Chan\AppData\Local\Google\Cloud SDK>
```

Figure 1: Google Cloud sdk (Full Size Image in Appendix A.3)

Python speech recognition:

The “free” option investigated using python's “speech_recognition”. In this module, the sound file types accepted are: WAV AIFF AIFF-C FLAC. Since the recording software used for testing recorded in m4a, a conversion software was necessary. After the clients’ feedback testing for emphasis was put on this option. However, when testing this module, it was found that its recognition of softer tones was weaker than the previous options. When trying the word “hey” softly, the software recognized it as “pilot”.

Code:

```
import speech_recognition as sr

r = sr.Recognizer()

with sr.AudioFile('D:/Users/Ethan Chan/AppData/Local/Google/Cloud SDK/soundfile.flac') as source:
    audio = r.record(source)

a = r.recognize_google(audio, language='en-US')
```

```
print(a)
```

3.1.4. Testing and Specifications

3.1.4.1. Main and Portable Unit

Although, there are 2 different pseudocodes for the main and portable units they were tested together due to how dependent they are. Since the prototypes were more theoretical in nature, they were tested by running them through different scenarios. This ensured that they would run smoothly in cohesion and effectively in varying situations. All of the conditions and test results have been compiled into **Table 2.**

Test Number	**Test Conditions**	**Test Results**
1	The main unit does not connect to the server.	If the main unit does not connect to the server, then, based on the pseudocode: the unit will attempt to connect three times, if the problem persists, then, the large indicator on the unit will turn red and the small indicators will flash red. Therefore, it passes the test.
2	Both units connect to the server, but for some reason a signal can't be sent to the portable unit.	If the signal isn't sent to the portable unit, the main unit will then attempt to send it a max of three more times. If the problem persists then the main unit will check to see if the unit is still connected to the server. If not, then turn the large indicator red and the small indicator a flashing red. Therefore, it passes the test.
3	The main unit successfully sends a signal to the portable unit, but the vibration keeps increasing and doesn't stop.	If the vibrations keep increasing and do not stop, the current model doesn't account for this, i.e. there is no fail safe as far as the software is concerned. (there is a reset button on the portable device itself, but that doesn't apply here) Therefore, it fails the test.

Table 2: Main and Portable Unit Testing

Team 2 conducted three test cases on the pseudocodes, each case gave a unique outcome. The test conditions were mostly situational and hypothetical in nature. The team decided to have hypothetical test conditions because of how the prototype is theoretical. The first condition was that the main unit did not connect to the server, this simulates having internet connection problems. This is an important issue due to how this device is integral to Fran's safety and wellbeing. It is important that the devices are either connected at all times, or that the user is always aware of the connection status. The next condition relates to an error in the devices themselves, as the team is developing a new product there is a possibility of the system registering false positives in terms of connection status. At the moment the device does a good job of informing its user about its state but does not troubleshoot complex issues. Finally, the last test condition investigates a problem in the current portable unit design. The device does not include a fail safe for mechanical errors, such as it is increasing in vibration with no limit. This is a place where improvement is necessarily in the next iteration of the prototype.

3.1.4.2. API

The more important aspect of this part of the project remains to be completed. This being the testing using the clients' voice. In the near future, the group will receive audio clips from the client to run through the speech-to-text software. This being, to observe the accuracy of these softwares with Fran's weaker voice. In place of this, the group used recordings of their own voices in place. For some recordings, the distance between the device and client was simulated, and for others the muffling of the voice was simulated. 4 sound levels were used per decibel level in this testing and the results are shown below (**Table 3**).

Decibels	Google	Python
30	100%	75%
40	100%	75%
50	100%	100%
60	100%	100%
70	100%	100%

Table 3: Speech to Text Testing Results at Different Decibels

As shown in the table, Google's version of the software was much more accurate. However, due to Google's fees, the python version will still be used as long as future testing allows. For now, the 75% margin at 40db and below was decided to be acceptable.

3.2 Hardware

The main focus of the hardware prototypes was to design prototypes that are small and compact but fulfill their necessary tasks. The main units design took into account a lot of the needs of the project. There was a focus put on fitting in a powerful enough light for Fran to be alerted that help is on the way. Refer to **Appendix A.4** for photos. The secondary unit was designed in order to be small enough that the staff easily be able to have it with them at all times but will still easily fit all the required components. The second unit was also designed in a manner to allow for the unit to be taken apart if needed to change the battery, as well as leaving ample space to fit in a charging port if a rechargeable battery is used. Refer to A**ppendix A.5** for large size photos.

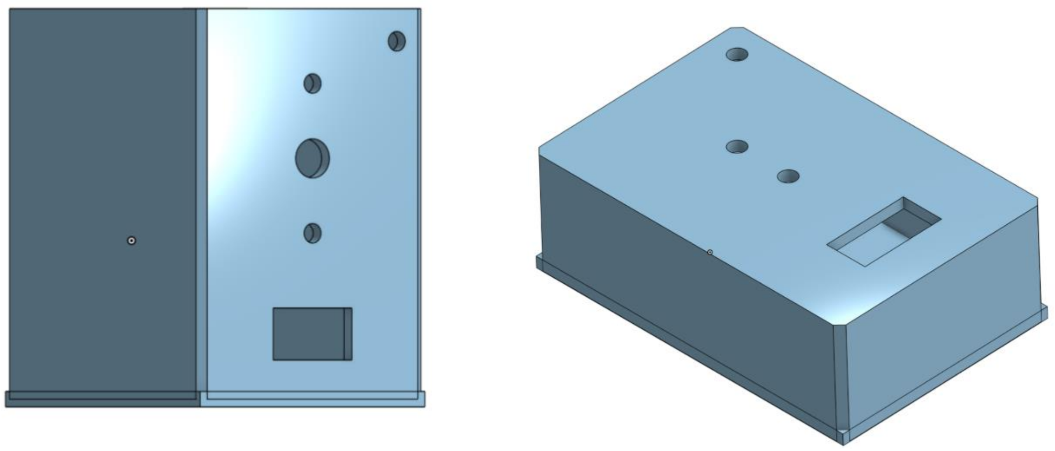


Figure 2: Main Unit (Left) and Portable Unit (Right) Models

3.2.1 Testing and Specifications

Since the hardware prototype was made to model what the 3D design would look like, it was not subjected to any particular testing. However as described above, specifications were kept in mind when creating the model.

2 Bill of Materials

An item was added to the BOM for every feature that had been asked for by the client. Not a single item was added as an "extra feature", for example, the main unit requires a microphone in order to pick up audio from Fran; LEDs are necessary for the indicators on both units, this ensures that the client is aware of the status of the machine at every step. Due to how each entry into **Table 4** is the bare minimum needed to fulfill each specification, Team 2 believes that it is both thorough and reasonable given the scope of their project. Furthermore, the BOM does not exceed the $100 CAD limit imposed upon them.

Retailer	**Item name & description**	**Cost per Unit (CAD$)**	**Number of Items**	**Total Cost (CAD$)**
Makerstore	Microphone	9.00	1	9.00
Alessandro	Vibrating Motor	Free	2	Free
Makerstore	PCB Mount Mini Speaker	4.00	1	4.00
Ethan	Battery Connector	0	1	0
Makerstore	LEDs	0.60	5	3.00
Makerstore	Protoboard	--	1	--

Walmart	Batteries (9V)	4.98	1	4.98
Makerstore	Resistors	--	3	--
CanaKit	Raspberry Pi Zero W board	12.95	1	12.95
CanaKit	Raspberry Pi Zero W Kit	32.95	1	32.95
CanaKit	Sd card	9.00	1	9.00
Makerstore	Buttons (Tactile button switch)	5.00	3	14
	Total Cost			**89.88**

3 Project Plan

The project plan was updated in order to include a detailed schedule of the next two weeks. The next two deliverables as well as prototype 2 have been split up into manageable tasks which were then distributed based on people's skills and their availability. The team's TA/PM did not mention any missing tasks, task responsibilities, milestones, or dependencies, in the feedback received from previous deliverables so no tasks were added. Refer to the **Project Plan Update 10-08-20** PDF for images of the project plan.

4 Conclusion

Meeting with Fran and her staff not only allowed Team 2 to gain further clarity on the project, but also helped them learn about potential improvements needed in the design. Based on the feedback the team was then able to curate a list of critical assumptions in their design. This allowed them to create their first prototype which has been split into 2 types: hardware and software. The software prototype consisted of light coding and pseudocode, this was then tested against potential situations the device may encounter, finding it to be a fairly effective first prototype. The hardware prototype consisted of 3d modeling of the main and portable units, this allowed them to clearly visualize what the system will look like. Due to its nature, it was not tested but fulfilled its purpose of providing the team with clarity about the devices physical design. Finally, based on the prototypes and client feedback a bill of materials was made. The BOM was chosen in a manner that covered all of the specifications without exceeding the $100 CAD limit. Prototype I allowed Team 2 to further their design process and learn more about the device they are making. This allowed them to detail their project plan, leaving them prepared and optimistic for the days to come.

5 Bibliography

[1] C. Gartenberg, "I made my own portable phone charger, and I didn't electrocute myself," *The Verge*, 03-Aug-2017. [Online]. Available: https://www.theverge.com/circuitbreaker/2017/8/3/16085856/diy-gadget-portable-phone-charger-car-altoid-tin-9-volt-battery-usb. [Accessed: 09-Oct-2020].

[2] MakerStore, "Electronics, Materials, and Merch," *MakerStore*. [Online]. Available: https://makerstore.ca/shop?keywords=raspberry pi&olsPage=search. [Accessed: 09-Oct-2020].

[3] "Poweradd EnergyCell 10000mAh Portable Charger Ultra-Compact Size Power Bank with Fast-Charging External Battery Pack Compatible with iPhone, iPad, Samsung, Huawei and More-Black," *Amazon.ca: Cell Phones & Accessories*. [Online]. Available: https://www.amazon.ca/Poweradd-EnergyCell-Portable-compatible-Smartphone/dp/B07MC33YG3/ref=sr_1_30?dchild=1&keywords=battery pack&qid=1602033645&sr=8-30. [Accessed: 09-Oct-2020].

[4] "Poweradd EnergyCell 5000 Power Bank, Mini Small Portable Charger with High-Speed Charging, External Battery Pack Compatible with iPhone, Samsung Galaxy, Pixel and More - White," *Amazon.ca: Cell Phones & Accessories*. [Online]. Available: https://www.amazon.ca/Poweradd-EnergyCell-Portable-High-Speed-Compatible/dp/B085XKFVYT/ref=sr_1_76?dchild=1&keywords=battery pack&qid=1602033645&sr=8-76. [Accessed: 09-Oct-2020].

[5] "Raspberry Pi Zero W (Wireless)," *CanaKit*. [Online]. Available: https://www.canakit.com/raspberry-pi-zero-wireless.html. [Accessed: 09-Oct-2020].

Appendix A

Appendix A.1: Main Unit Pseudocode

Variables:
systemcount - This counter counts the number of times the system has tried to connect to the server.
Flag (0 - 5) - Each of these flags is used to see if a condition has been met.
count - This counter counts the number of times the system has tried to connect to a server
count2 - This counter functions the same as count.

Global Variables: systemcount

Call **Connection Function**

Server Function:

```
systemcount = systemcount + 1
While systemcount < 3:
    While Flag = False and count < 3:
        Check if connected to portable unit:
            If yes:
                Call Audio function
            If not:
                count = count + 1
                Flag = False
Turn large indicator red
Red small indicator beings blinking every 5 seconds
```

Audio Function:

```
While Flag1 = False
    Check if there is noise:
        If yes:
            Is the noise = "hey" or "help"?
                If yes:
                    Call Server Function
                If not:
                    Flag1 = False
        If not:
            Flag1 = False
```

Connection Function:

```
While Flag2 = False and count2 < 3:
    send signal to portable unit
        If successful:
            Turn large indicator yellow
            Flag2 = True
            Call Confirm Function

        If not:
            Large indicator stays yellow and blinks every 15 seconds
```

```
                    count2 = count2 + 1
                    Flag2 = False
Turn large indicator red
Call **Server Function**
```

Confirm Function:

```
While Flag3 = False:
        Check if the main unit receives the “ok signal”:
                    If yes:
                            Turn large indicator green
                            Call **Arrived Function**
                    If not:
                            Flag3 = False
```

Arrived Function:

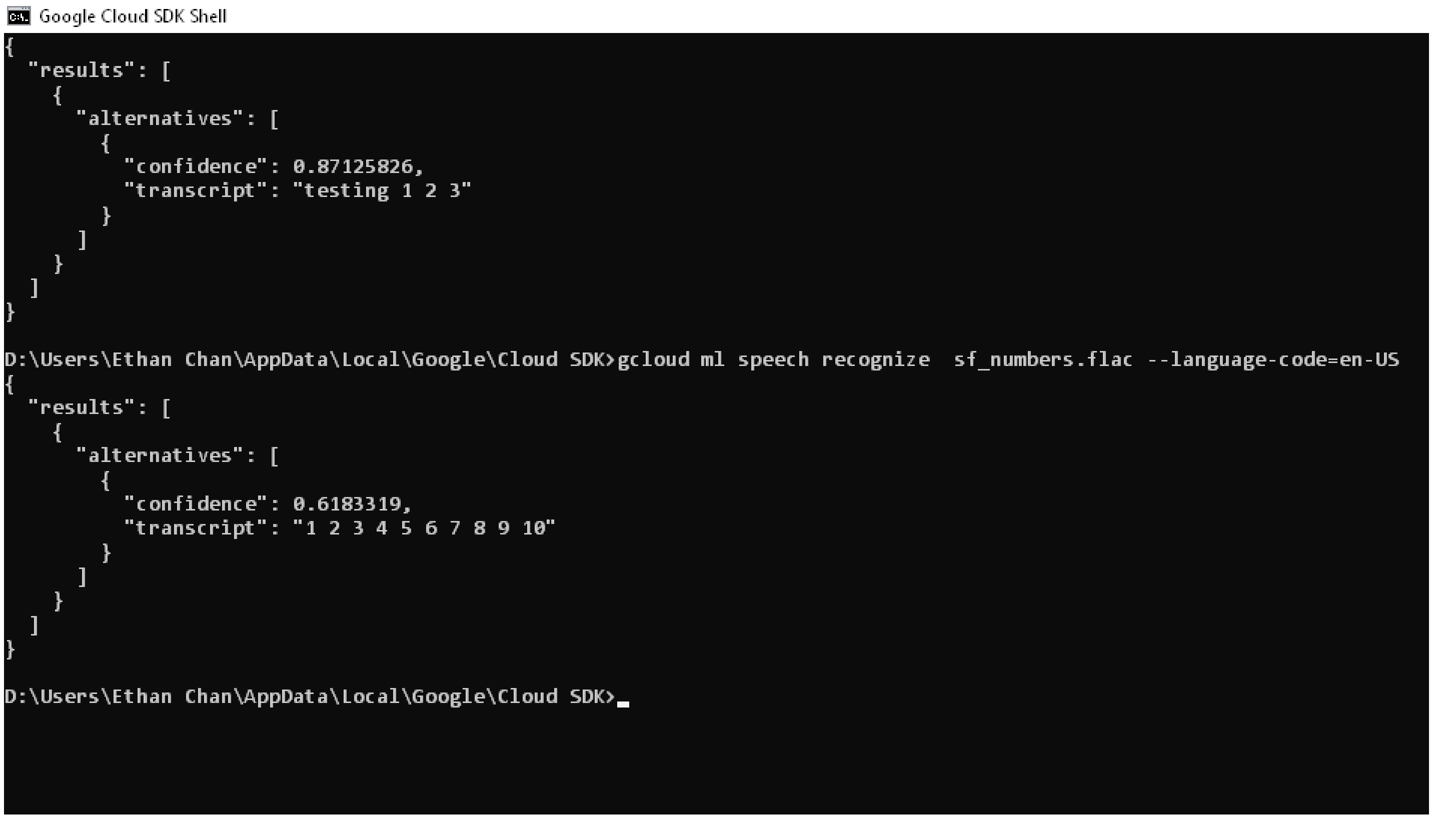
```
While Flag4 = False:
        Check if the “here” button has been pushed:
                If yes:
                        Send “Stop signal” to portable unit
                        systemcount = 0
                        Call **Server Function**
                If not:
                        Flag4 = False
```

Appendix A.2: Portable Unit Pseudocode

Variables:
connectioncount - a counter used to trigger the motor after 3 minutes of disconnection
Motor1 - the trigger for the motor to vibrate
situation1-3 each of theses check to see if condition has been met
safecount - a check to see how long the call has been sent to the portable unit
Audiocount - same as the safecount, another condition to be met.
Sound - triggers the audio chime
Indicator- triggers the LED light to a specific colour

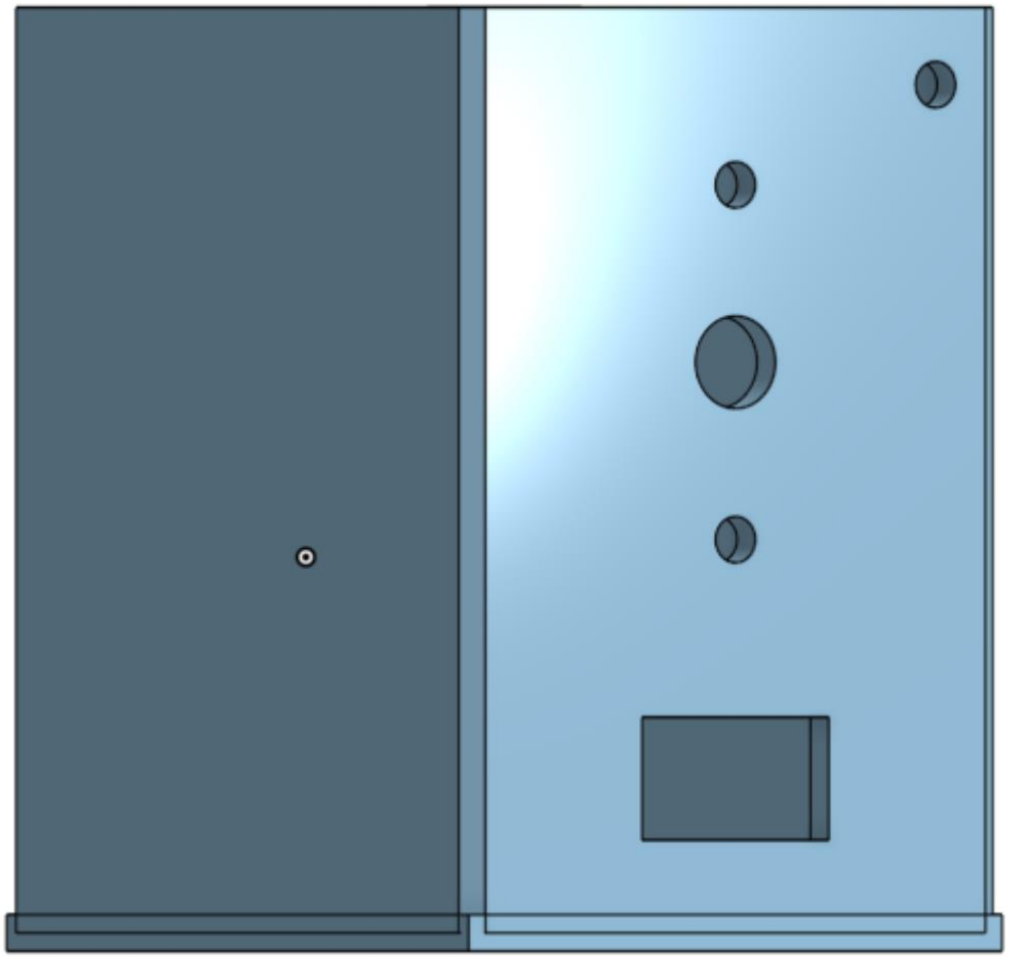
```
Call Connection Function

Connection Function
Check if connected to main unit
    If yes
        Call Server Function
    If not
        Red indicator stars blinking every 5 seconds
        connectioncount++
            If connectioncount > 180
                turn motor1 on.


Server Function
While Situation1 == false
            Check if signal has been sent.
                If yes
                    Vibrate “motor1” & turn indicator flash green
                If not
                    Situation 1 = false.


Confirm Function
While Situation3 == false
    Check if “ok button” has been pressed
        If yes
            Send “ok signal” to main unit
        If not
            Call Neglect Function
            Situation3 = false


Neglect Function
Check if “Stop signal” received:
    While Situation 2 == false

        If “ok button” & “Stop signal” == false
            safecount = safecount + 1
        audiocount = safecount/3
        If safecount > 3 and audiocount < 3
                Increase motor1 by 1 factor
        Else
                Play sound


Else if “Stop signal” == false and “ok button” == true
        safecount = safecount + 1
        audiocount = safecount/3
        If safecount > 3 and audiocount < 3
                Increase motor1 by 1 factor
        Else
                Play sound

Else if “Stop signal” == true and “ok button” == false
        safecount = safecount + 1
        audiocount = safecount/3
        If safecount > 3 and audiocount < 3
                Increase motor1 by 1 factor
        Else
                Play sound

Else if “ok button” & “Stop signal” == true
        Stop Motor1, sound and indicator
        Situation2 = true
        Call Connection Function
```

Appendix A.3: API Code


Google Cloud SDK Shell

```
{
  "results": [
    {
      "alternatives": [
        {
          "confidence": 0.87125826,
          "transcript": "testing 1 2 3"
        }
      ]
    }
  ]
}

D:\Users\Ethan Chan\AppData\Local\Google\Cloud SDK>gcloud ml speech recognize  sf_numbers.flac --language-code=en-US
{
  "results": [
    {
      "alternatives": [
        {
          "confidence": 0.6183319,
          "transcript": "1 2 3 4 5 6 7 8 9 10"
        }
      ]
    }
  ]
}

D:\Users\Ethan Chan\AppData\Local\Google\Cloud SDK>
```

Appendix A.4: Main Unit Model



Appendix A.5: Portable Unit Model