GNG2101

Design Project User and Product Manual

SpeakEasy Communication Device

Submitted by:

A02-COM1

Omar Nasr, 300365142

Emily Rudderham, 300137648

Robert Gauvreau, 300388180

Ahmad Muslat, 300292034

David Rosocha, 300366191

Callan Underwood, 300152504

December 3, 2024

University of Ottawa

Table of Contents

Table of Contents
List of Tables
List of Acronyms and Glossary
1 Introduction
2 Overview
2.1 Cautions & Warnings
3 Getting started
3.1 Configuration Considerations
3.1.1 Physical Configuration
3.1.2 Electrical Configuration
3.1.3 Software Configuration
3.2 User Access Considerations
3.3 Accessing/setting up the System
3.3.1 Physical Set-Up
3.3.2 Electronics Set-Up
3.3.3 Software Set-Up
3.4 System Organization & Navigation
3.4.1 Exterior components
3.5 Exiting the System
3.5.1 Putting Away the CA
4 Using the System
4.1 SpeakEasy Enclosure
4.1.1 Input Buttons
4.1.2 Charging Port
4.2 SpeakEasy Application
5 Troubleshooting & Support
5.1 Error Messages or Behaviors
5.2 Maintenance
5.3 Support
6 Product Documentation
6.1 Electrical Enclosure
6.1.1 BOM (Bill of Materials)
6.1.2 Equipment list
6.1.3 Instructions
6.2 Wheelchair Mount
6.2.1 BOM (Bill of Materials)
6.2.2 Equipment list
6.2.3 Instructions
6.3 Communication Array Circuit
6.3.1 Bill of Materials for Electrical Component
6.3.2 Equipment list
6.3.3 Instructions
6.4 Application
6.4.1 Bill of Materials for the Software Component
6.4.2 Equipment list
6.4.3 Instructions
6.5 Testing & Validation
6.5.1 Electrical Testing

6.5.2 Mechanical Testing
6.5.3 Software Testing
Part 1 | Arduino Software Testing
Part 2 | Arduino Software Testing
7 Conclusions and Recommendations for Future Work
APPENDICES
8 APPENDIX I: Design Files
9 APPENDIX II: Other Appendices

List of Tables

Table 1. Acronyms v
Table 2. Glossary v
Table 3. Referenced Documents 37

List of Acronyms and Glossary

Table 1. Acronyms

Acronym	Definition
CA	Communication array
PVC	Polyvinyl chloride
ABS	Acrylonitrile butadiene styrene
TPU	Thermoplastic polyurethane
MDF	Medium-density fibreboard
AAC	Augmentative and Alternative Communication (device)

Table 2. Glossary

Term	Acronym	Definition
Enclosure		Box that holds and protects buttons and electrical components
SpeakEasy		The SpeakEasy communication array.
TPU Lip		The 3D printed TPU lip mounted on the top edges of the device. The edgeguard

1 Introduction

This User and Product Manual (UPM) provides the information necessary for caretakers to effectively use the SpeakEasy and for prototype documentation. The SpeakEasy is an AAC designed for a non-verbal user with limited motor skills who is wheelchair-bound. This document contains – in the order listed – a basic overview of the product, simple set-up and removal instructions, step by step instructions for how to use each part of the product, support and troubleshooting for potential errors, and extensive product and prototyping documentation.

2 Overview

Recent studies show that up to 1% of the world population suffers a degree of speech, or communication need 1. These needs can be met using an Augmentative and Alternative Communication method which can include manual signs, picture communication boards and speech-generating devices.

The SpeakEasy is intended for users in need of a device to audibly communicate messages that can be operated with limited motor skills and from the user's wheelchair.

There exists a variety of speech generating devices on the market which accommodate users with limited motor skills. However, the prices range from the hundreds to even thousands of dollars for these devices, they often rely on caretakers to record their own voice, and they do not offer built-in wheelchair mounting.

Our solution to this is an affordable all-in-one package which offers an accessible communication device, a quick wheelchair mounting mechanism and a user-friendly application to update the devices prompts wirelessly.

Figure 1: Overview of SpeakEasy Communication Device

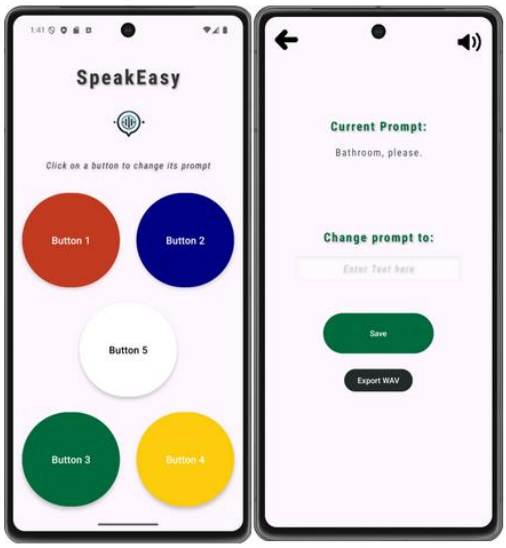

Figure 2: SpeakEasy Prompt Editing Application

The SpeakEasy communication device is built with a focus on accessibility, durability, and ease of use, providing a seamless experience for individuals with limited motor skills. The system attaches securely to a wheelchair using a clamp mechanism, which ensures a stable and reliable mount while allowing for quick adjustments or removal when needed. The enclosure, designed to house the internal components, is made from lightweight MDF.

The device features a long-lasting battery with over three months of continuous use and is also rechargeable for convenience. The user interface is simple and intuitive, with color-coded buttons for easy operation. Additionally, the SpeakEasy can be customized to suit individual preferences.

Each button on the device is linked to a specific audio message, offering clear and straightforward functionality. Inside the enclosure, a microcontroller serves as the brain of the system, processing input from the buttons and triggering the playback of pre-recorded audio prompts stored on a microSD card. A high-fidelity speaker ensures that the audio is clear and audible in various environments.

The system is complemented by a mobile application that connects via Wi-Fi, allowing caregivers to update the audio prompts easily. Through the app, caregivers can add, remove, or modify the prompts. This functionality enables personalized communication tailored to the user's specific needs.

Designed with practicality in mind, the device is compatible with a wide range of wheelchairs. The clamping mechanism adapts to various wheelchair designs and ensures that the device remains securely in place, even during movement. To maintain stability, caregivers are advised to periodically check the clamp for tightness. The SpeakEasy is also designed to be safe for everyday use, with insulated electronics to prevent electrical hazards and durable construction to withstand minor impacts or spills.

For a comprehensive view on how the software, refer to the diagram below.

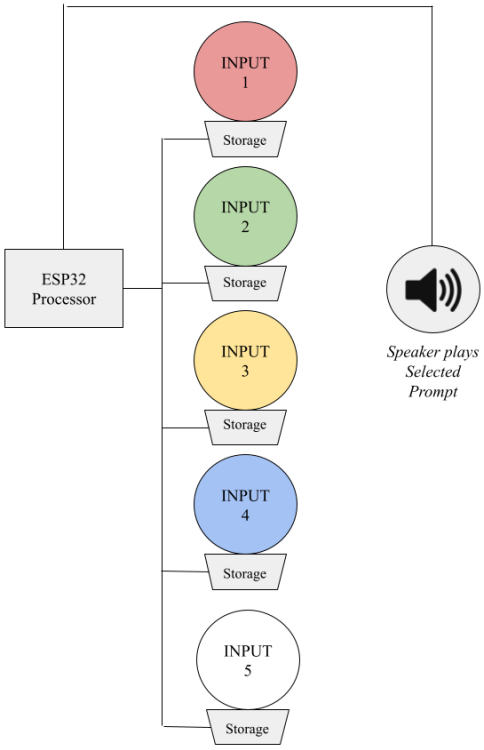

Figure 3: Flowchart of Communication Array

When a button is pressed, the audio file that it correlates to is sent to the ESP32 processor and it decodes the WAV file into sound, that is then sent to the amplifier of the speaker to play the prompt.

2.1 Cautions & Warnings

- Do not lean on the device.
- Do not rest objects on the device.
- Keep water and other liquids away from the device.
- Confirm proper mounting and clamp locking before using the device.
- If the device has been mounted for a long period of time, confirm clamp tightness (Use and vibration may loosen the mount over time).
- Avoid applying excessive force on the buttons.
- Do not insert the charging port while using the device, the device does not work while charging.

3 Getting started

1. Unplug the charging port from the device
2. Connect you phone to the ESP32E Wi-Fi to view the current prompts in the app.

3.1 Configuration Considerations

3.1.1 Physical Configuration

The exterior components of the CA are configured by holding the pipe with the two O-clamps at a vertical and clamping the two O-clamps onto the wheelchair's transfer bar. The transfer bar is the U-shaped bar under the armrest. Once this is done and the software configuration is complete, the CA is ready for use.

Figure 4: Wheelchair Mount Installation

3.1.2 Electrical Configuration

The battery is connected to the microcontroller's mini-USB port, converting the 5V pin of the microcontroller to a power output. Although the battery boasts a significant battery life, the opportunity to recharge the device is there to extend the life of the product and reduce the ecological footprint of buying and recycling battery cells. Looking at the enclosure from a birds-eye view, the speaker is located at the right most side of the wheelchair mount, connected to the audio amplifier.

At the leftmost side of the enclosure, a charging port can be found. The buttons themselves are not interconnected, meaning every button press is distinct. The prompt can be customized through an app, which connects to the onboard Wi-Fi chip found in the ESP32E microcontroller. Next to the microcontroller is a micro-SD card reader, which stores the audio files to be played by pressing the buttons.

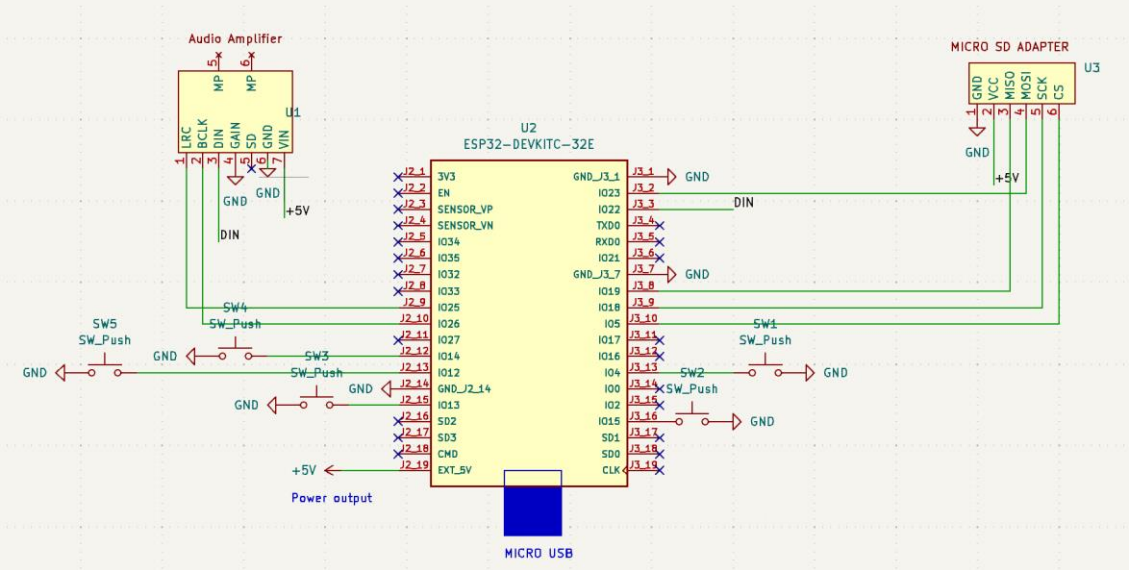

Figure 5: Circuit Diagram

3.1.3 Software Configuration

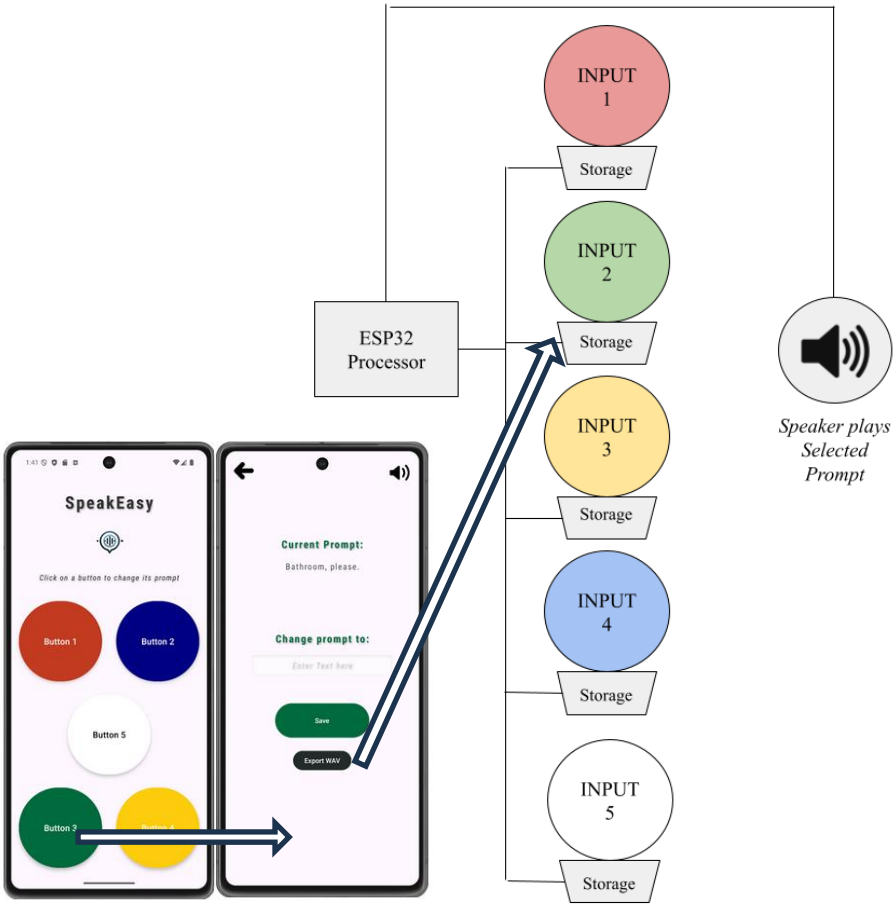

Figure 6: Software Configuration Diagram

To demonstrate the configuration of the software, let's demonstrate an example where the client wants to change the prompt of the green button. Firstly, the client will begin by clicking on the green button on their mobile app, which will then open a new page where they will be able to access the current prompt and will have the opportunity to change the prompt to whatever they would like. By entering text into the text box, pressing save, and then exporting, a WAV file will be sent wirelessly (given that the app is connected to the ESP-32's Wi-Fi network) to the storage corresponding to the green button. After this, when the user presses the green button, the ESP32 microprocessor will fetch the file, decode it, and send the information to the speaker to play the sound.

3.2 User Access Considerations

This product is designed for a non-verbal user with limited motor skills, specifically cerebral palsy. The device is designed specifically for wheelchair users, however the wheelchair mounting system can be removed via the 2 bolts underneath the enclosure if desired. While we found great success with the users' ability to reliably hit a desired button, users with other mobility conditions such as Parkinson's may find the button inputs challenging to use.

3.3 Accessing/setting up the System

3.3.1 Physical Set-Up

1. If the O-clamps are not already open, loosen the wing-head screws on the clamps until they can be pushed upwards to unlatch the clamp.
2. Bring CA to wheelchair and align clamps to top and bottom of transfer bar.
3. Press the wing-head screws down to lock the O-clamps in place.

3.3.2 Electronics Set-Up

Using the Communication Array

1. Apply pressure to one of the five buttons to initiate a prompt.

Charging the Device

1. Ensure that the device will be receiving a minimum of 10 Watts of power.
2. Connect the charger to the installed charging port in the device)

NOTE: You can verify that the device is being charged by pressing a button. If it is successfully charging, a prompt from any of the 4 ports will not be activated.

3.3.3 Software Set-Up

The software setup for the SpeakEasy communication device has been designed to be straightforward and user-friendly, ensuring ease of use for caregivers and users alike. At the core of the system is a mobile application that connects to the device via Wi-Fi. This app serves as the main interface for customizing the audio prompts linked to the device's buttons. Setting up the software begins with connecting your smartphone or tablet to the device's Wi-Fi network called ESP32-Audio, with the password "12345678". Once connected, the app automatically detects the device and syncs its current settings, displaying the prompts assigned to each button. Note, that we would manually download the app onto the client's phone as it costs money to publish to the play store. However, for future development, if we were to publish our app online, then it would be independently accessible on the Google Play Store for free.

To modify the prompts, the caregiver uses the app to input new text for any of the buttons. The app then processes this input using built-in Text-to-Speech (TTS) functionality, generating high-quality MP3 audio files. These files are stored on the device's microSD card, ensuring they are ready for playback when the corresponding button is pressed. The app also allows the caregiver to preview the prompts by playing back the TTS audio before finalizing any changes, providing an opportunity to confirm the accuracy and clarity of the message.

The app is designed to prioritize simplicity and responsiveness, with clear navigation and minimal steps required to perform updates. This ensures that even caregivers with limited technical experience can comfortably manage the device. The software also includes built-in safeguards to prevent issues such as duplicate file names or incomplete uploads, ensuring that the prompts are correctly stored and accessible on the device.

3.4 System Organization & Navigation

3.4.1 Exterior components

The exterior components of the CA include the buttons, the enclosure which houses the electronics, the ABS tube and fittings which hold the enclosure and the two O-clamps which mount onto the transfer bar of the wheelchair.

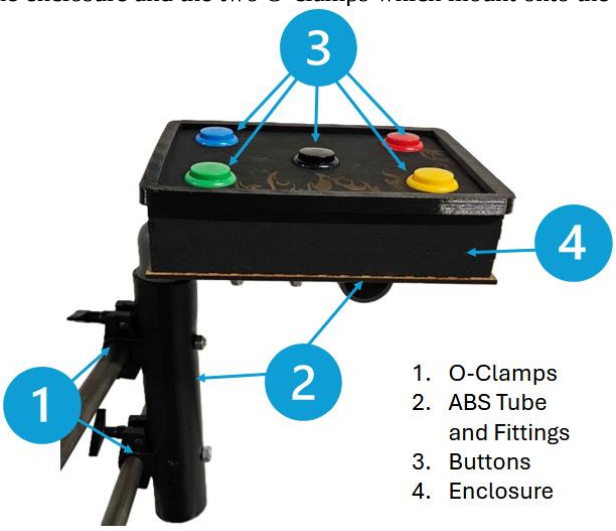

Figure 7: Overview of External Components

The bottom piece of the enclosure has two screw holes which align with the two holes drilled through the horizontal ABS tube. Two screws are fed through the base of the enclosure and the ABS tube to mount the enclosure onto the ABS fixture. A locking nut and washers are used to keep these screws in place.

Two cut-outs and two screw holes are made in the vertical piece of ABS. The O-clamps rest in the cut-outs and a screw is fed through the screw and through the screw hole on the opposite side of the tube to secure the clamps to the ABS. A locking nut is used to keep these screws in place.

The buttons are press-fit into holes created in the top of the enclosure.

3.5 Exiting the System

3.5.1 Putting Away the CA

1. To remove the CA from the wheelchair, loosen the wing-head screws on the clamps until they can be pushed upwards to unlatch the clamp.

2. Pull CA away from wheelchair.

4 Using the System

The following sub-sections provide detailed, step-by-step instructions on how to use the various functions or features of the SpeakEasy.

4.1 SpeakEasy Enclosure

The SpeakEasy enclosure consists of only two distinct parts with user input/ modulation. Turning on the SpeakEasy is not a worry, as the device is always turned on due to minimal power consumption.

4.1.1 Input Buttons

This section describes how to output a prompt using the input buttons.

1. Press down on the button whose prompt you want to output until an audible “click” noise is produced, and the button becomes depressed.

Note: Consult the SpeakEasy application (instructions on how to do so in section 4.2) to view and change the prompts.

4.1.2 Charging Port

1. Locate the small rectangular hole on the side of the enclosure closest to the vertical ABS tube, shown below.

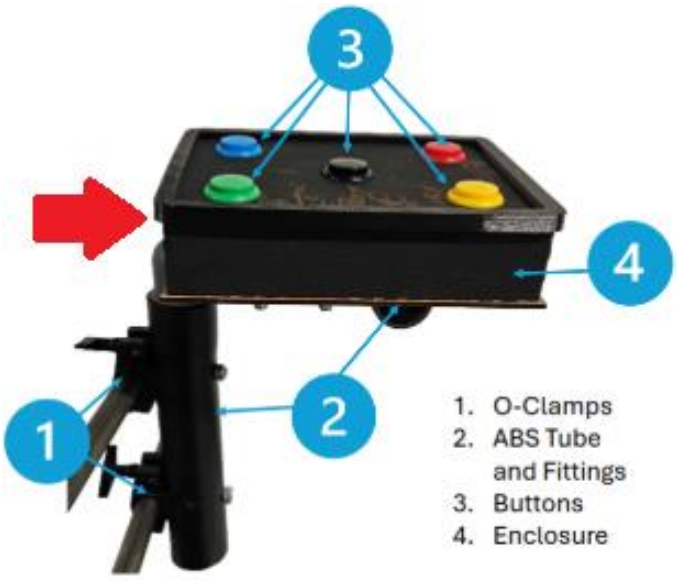

Figure 8: External Components with Charging Port Side Highlighted

2. Plug a USB charging cable into the port located inside the rectangular hole.
3. Plug the other end of the USB charging cable into an applicable power source to begin charging.

4.2 SpeakEasy Application

The SpeakEasy mobile application is the central hub for customizing and managing the functionality of the SpeakEasy communication device. It has been carefully designed with user accessibility and caregiver convenience in mind, ensuring that it is both easy to use and robust enough to handle all necessary interactions with the hardware. The app bridges the gap between the user and the device, allowing for real-time customization and seamless operation.

At its core, the SpeakEasy app provides an interface to manage the audio prompts associated with the device's buttons. Upon launching the app, users are greeted with a simple, intuitive interface that displays the current setup. The app connects to the SpeakEasy device via a secure Wi-Fi network, ensuring a reliable and private connection. Metrics from testing showed an average connection time of just 2 seconds, emphasizing the app's responsiveness and efficiency.

The home screen of the app is organized around the four main buttons of the SpeakEasy device, each represented as a tile or card on the screen. Selecting any tile opens a detailed editor for that button’s associated audio prompt. Caregivers can input new text directly into the editor, and the app processes this text through its built-in Text-to-Speech (TTS) engine. The TTS system converts the text into a high-quality MP3 file, with a tested generation time averaging 1.8 seconds per file. This speed ensures that caregivers can quickly update prompts without causing frustration or delays.

The app also includes a preview feature, allowing caregivers to play back the generated audio directly from their phone before uploading it to the device. This step is crucial for verifying clarity and correctness, especially in cases where the user relies on the device for critical communication needs. Once confirmed, the audio file is uploaded to the SpeakEasy device's microSD card, a process that takes approximately 2 seconds per file based on system testing.

Customization extends beyond just text-to-speech prompts. The app allows caregivers to assign color codes to each button within the app interface, mirroring the physical button colors on the device. This visual connection enhances usability by ensuring that both the user and caregiver can easily associate each button with its corresponding function.

The app is designed to support up to five active prompts at a time, aligning with the five physical buttons on the device. While this might appear limiting compared to more complex systems, it is intentionally kept simple to ensure accessibility for individuals with cognitive or fine motor skill incapabilities. Despite the simplicity, the appllications robust data handling ensures that each MP3 file is stored efficiently, with average file sizes of just 275 KB. This compact size allows the device to hold additional files for future expansion, should a caregiver wish to swap prompts in and out as needed.

The app also incorporates built-in safeguards to prevent errors during customization. For example, if an MP3 file fails to upload due to connectivity issues, the app alerts the caregiver and provides clear instructions for retrying. Similarly, file naming conventions are enforced automatically to avoid overwriting existing prompts or creating conflicts on the device's storage.

Navigation within the app is smooth and responsive, with a measured navigation latency of 0.3 seconds per action during usability tests. The app also features a help section with step-by-step instructions and troubleshooting tips, ensuring that caregivers can resolve most issues independently without needing external support. During user satisfaction surveys, caregivers rated the app an average of 9.3 out of 10, citing ease of use, speed, and clarity as standout features.

In addition to managing prompts, the SpeakEasy app logs the last changes made, providing a basic history for caregivers to reference. This feature is especially helpful in shared caregiving scenarios where multiple people might interact with the device. Security is also a priority; the app requires a unique pairing with the SpeakEasy device during initial setup to ensure unauthorized users cannot access or modify the prompts.

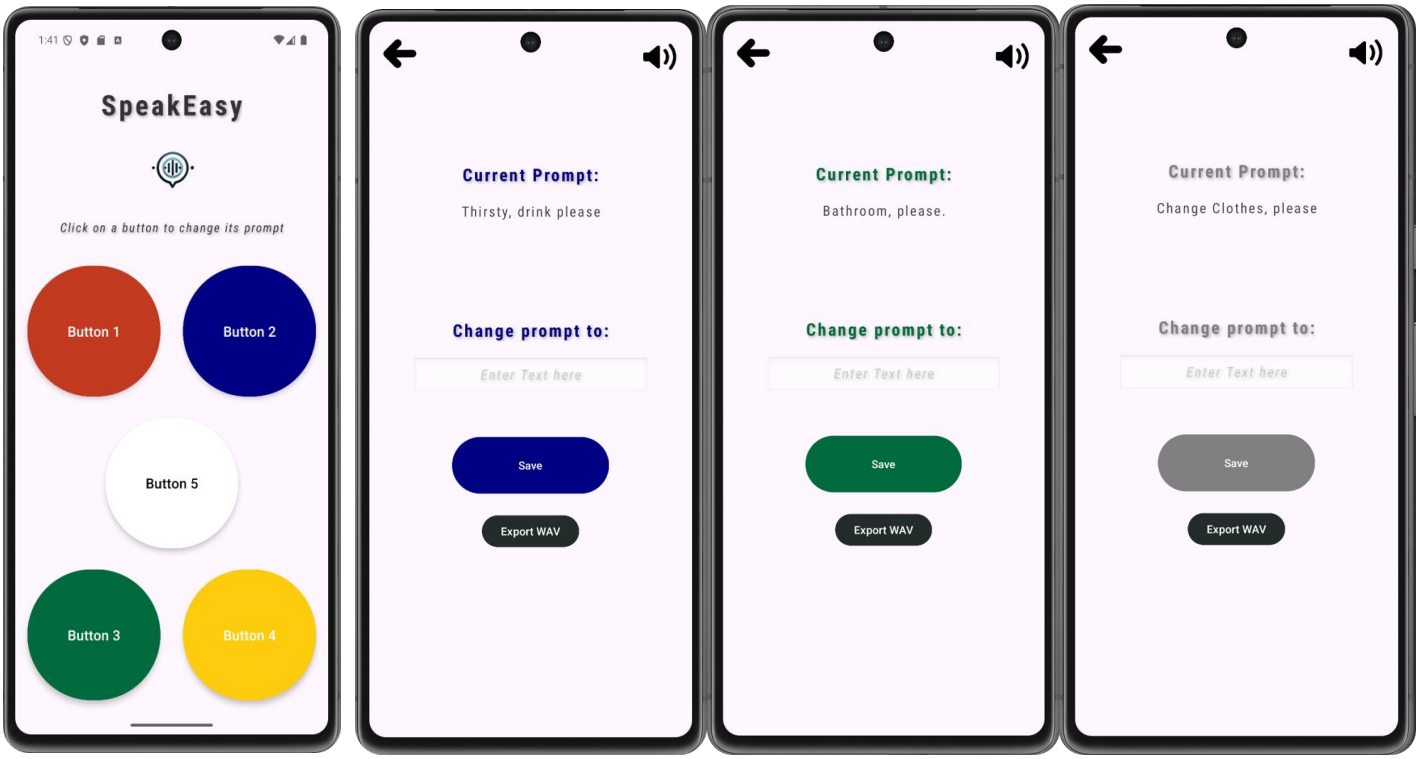

Figure 9: Various Screenshots of the SpeakEasy Application

5 Troubleshooting & Support

5.1 Error Messages or Behaviors

No error messages or issues with the software should ever appear given the amount of time we worked on it. Possible issues that may arise could be that the battery is dead and needs to be recharged, a wire has broken off or come loose and lastly that a wire has been short circuited with another wire. The last 2 issues that I mentioned are for awareness but there is a slim-to-none chance of this occurring due to carefully soldering and adding electrical tape to every conductive wire to avoid short circuits. If for whatever reason, a wire has short circuited itself, the battery knows this and turns itself off. The issue must then be corrected, and the battery would need to be powered on again.

5.2 Maintenance

1. Once a month, all nuts should be tightened by hand to ensure they remain secure.
2. With frequent daily use, the battery should be charged every 2-months to prevent it from running out of charge. Consult Section X for instructions.

5.3 Support

In any case where emergency assistance or system support is required, please contact onasr047@uottawa.ca with a detailed description of the issue. We will follow up with the next steps to take, after the problem is identified.

6 Product Documentation

6.1 Electrical Enclosure

6.1.1 BOM (Bill of Materials)

Table 3: Bill of Materials for Electrical Enclosure

Part	Description	Quantity	Cost per unit	Link
MDF	12” x 24” x 1/8” thickness MDF board	1	C$2.50	Makerstore
TPU	3D printer filament. 1.75mm 85A Ninjaflex TPU	30 g	C$70.99 / 500g	3D Printing Canada
Acrylic Paint	Black Acrylic Paint	15 ml	$9.99 /100 ml	Amazon

6.1.2 Equipment list

Table 4: Equipment List for Electrical Enclosure

Equipment	Purpose	Alternative
Laser Cutter	To cut out the sides of the enclosure	
FLSUN 3D Printer	To print the edge guard	Any 3D printer capable of printing TPU
File	To file down the laser cut sides until they fit together	Sandpaper
Painting Brush	To apply acrylic paint onto MDF before laser cutting	
Mill	To bore out two holes where the enclosure is screwed onto the mount.	Drill Press / Hand Drill

6.1.3 Instructions

6.1.3.1 Overview

Enclosure:

1. Prepare the enclosure laser cutting file using Inkscape
    a. Ensure correct dimensions
    b. Ensure correct line thicknesses.
2. Paint the MDF board to be used
3. Laser cut the painted MDF board

4. Assemble the cut-outs, file down interfacing tabs if necessary

Edgeguard:

1. Prepare 3D printer for TPU printing.
2. Access the edgeguard's file on CURA (or the appropriate software for printers other than the FLSUN).
3. Confirm that the printing settings are appropriate.
4. Start Print.

6.1.3.2 Detailed Instructions

Enclosure:

1. Prepare the enclosure laser cutting file using Inkscape

The laser-cutting file for the enclosure is readily available on MakerRepo under the repository named "SpeakEasy | GNG2101". The file is titled "Enclosure_LaserCut_V6.pdf". This file is ready for laser cutting and can be directly used, but for any edits to the dimensions or other aspects of the enclosure, the file titled "Enclosure_LaserCut_V6.svg" can be directly opened in Inkscape and edited.

2. Paint the MDF board to be used

Painting the MDF board is optional, but highly recommended. The paint provides protection against moisture and prevents splintering of the MDF. The specific color of the paint itself does not make a difference. When painting the board, it is recommended to only stroke in a single orientation (horizontally or vertically) to improve the final aesthetic of the product.

3. Laser cut the painted MDF board

Upload the laser-cut-ready file to the laser cutter. Select the recommended settings for the specific device in question. For the Epilog Laser Helix 24''×18'' 60W, the following setting should be used:

- Rastering
    - 25% Power
    - 100% Speed
- Vectoring
    - 100% Power
    - 25% Speed

After selecting the settings, ensure that there is sufficient space on the board and in the laser-cutter tray for the necessary cuts, activate any pumps/filters that the laser cutter uses, then press start.

4. Assemble the cut-outs, file down interfacing tabs if necessary

Push out the cut-outs from the MDF board, then check if they fit together. If they fit together, then assemble the enclosure. If they do not fit together, then file down the tabs that interface slightly and try again. Do so until all the sides fit tightly together. Be careful as to not make the sides loose.

Edgeguard

1. Prepare 3D printer for TPU printing.

Install TPU filament then heat the printer nozzle to 220 C.

2. Access the edgeguard's file on CURA (or the appropriate software for printers other than the FLSUN).

The G-code file for the edge guard is available on the MakerRepo repository SpeakEasy | GNG2101, titled "FV_P6 - EdgeGuard(square).gcode".

3. Confirm that the printing settings are appropriate

The Solidworks file is also available in the repository. It can be changed as necessary.

4. Start Print

Please reference 6.3.3 below for instructions on how to complete the installation of the electrical components.

6.2 Wheelchair Mount

6.2.1 BOM (Bill of Materials)

Table 5: Bill of Materials for Wheelchair Mount

Part	Description	Quantity	Cost per unit	Link
ABS Pipe	1.5" diameter x 3' length ABS Pipe	1	7.48	Home Depot
Pipe Elbow	1.5" diameter 90 degree Poly Pipe elbow	1	1.86	Home Depot
Pipe Tee	1.5" diameter Poly Pipe Tee	1	3.11	Home Depot
O-Clamp	Fits 3/4 inch / 20mm truss	2	8.00	AV Shop

Bolt	5/16” diameter by 2 1/2” length hex head cap screw	2	1.19	Home Depot
Locking nut	5/16” locking nut	2	1.33	Home Depot

Note: The specific tube material and fittings recommended in the table are not required to create the wheelchair mount. Any material which can form an L-shape, can reliably support 5 pounds and can accommodate cutouts and drill holes for the clamps and screws is suitable for this application.

6.2.2 Equipment list

Table 6: Equipment List for Wheelchair Mount

Equipment	Purpose	Alternative
Heat gun	Heat ABS and pipe fittings to assemble them together	
Dremel (cut-out wheel)	To make cut-outs for clamps in ABS pipe	
Band saw (or hacksaw)	To cut ABS pipe to desired length	Hacksaw
Mill	To create holes in ABS for rotary tool to cut between to make cut-outs To make holes for clamp screws and enclosure screws to go through	Drill press or drill
Mallet	Hammer the ABS pipe and fittings together	Hammer
File	To file down the cut-outs to fit the O-clamp	Sandpaper

6.2.3 Instructions

6.2.3.1 Overview

5. Cut ABS pipe to correct length.
    a. Vertical piece
    b. Horizontal piece
6. Insert 90-degree elbow and Tee fittings into ABS pipe pieces.
7. Drill four holes in ABS pipe for each of two clamp cut-outs.
8. Connect holes using a cutting blade on a rotary tool.
9. Drill a hole in each cut-out for clamp screw.

6.2.3.2 Detailed Instructions

Step 1: Cut ABS Pipe to Correct Length

The mounting for the CA is designed to have the buttons level with the top of the armrest. As such, the correct length of ABS pipe depends on the wheelchair model and the configuration of the armrest. Descriptions of the measurements to be gathered are shown in the table below.

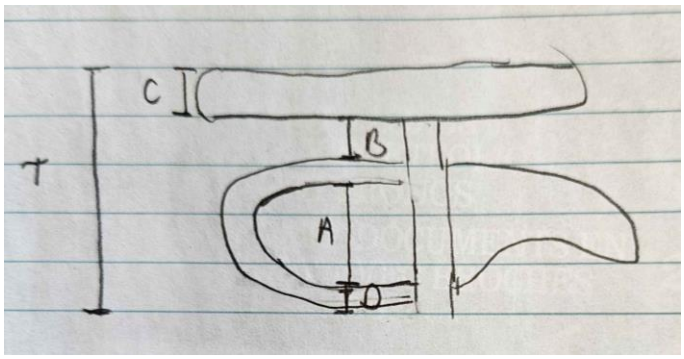

Figure 10: Diagram of Wheelchair Measurements

Table 7: Glossary of Wheelchair Measurements

Label	Measurement	Quickie SR45
A	Distance from bottom edge of top portion of the transfer bar and top edge of bottom portion of transfer bar	3.75”
B	Distance from bottom of armrest to top of transfer bar	1.66”
C	Thickness of armrest	1.33”
D	Diameter of transfer bar pipe	0.80”
T	Distance from top of armrest to bottom of transfer bar (T = C + B + A + 2*D)	8.35”

Once these measurements are taken, the length of the two ABS pieces are calculated as follows:

Vertical ABS Length = T – (Height of enclosure)– (Diameter of 90-degree elbow) + (1 inch extra)

Horizontal ABS Length = (Length of enclosure) - (Diameter of Tee fitting)

Note: An additional inch is added to the length of the vertical pipe to ensure there is sufficient material around the cut-outs for the O-clamps.

Note: See Section X to obtain enclosure dimensions.

Step 2: Assemble Pipe and Fittings

Figure 11: Order of Connection of Fittings to ABS Tube

Once the two pieces of ABS pipe are cut to the right dimensions, they can be connected to the pipe fittings.

1. Firstly, the 90-degree elbow fitting must be inserted into the vertical ABS piece. To do this, clamp the longest ABS piece in a vice and heat it with a heat gun. After a couple of minutes, press the 90-degree elbow into the heated end of the ABS pipe. Using a mallet, hammer the 90-degree elbow until the ridge near its bend almost touches the ABS pipe.
2. Once these pieces cool, place the shorter piece of ABS into a vice, heat it with a heat gun, and insert the exposed end of the 90-degree elbow into the ABS. As with the previous connection, hammer the elbow until the ridge near its bend almost touches the ABS pipe.
3. Once the rest of the pieces have cooled, the Tee fitting can be added. Clamp the L fixture in a vice and heat the exposed end of the shorter piece of ABS. Finally, press in the Tee fitting and hammer until the ridge is close to touching the ABS.

Figure 12: Hole Placement and Dimensions for Steps 3.2 and 3.3

Step 3: Drill Holes and Cut-Outs

1. Using a mill or drill press, drill two 5/16-inch holes that are 2 inches apart through the horizontal ABS piece. This will be used to secure the enclosure to the wheelchair mount.
2. Next, using a mill or drill press, drill four holes around ¼ inches in diameter (screw size is not critical here). These holes must be positioned such that they form the corners of a rectangle which is 1.5 inches in length and 1 inch in height. This is the size of the rectangular extrusion of the O-clamps.
3. Use a rotary tool with a cutting blade to cut out a rectangle by connecting each set of four holes. If the clamp does not immediately fit after cutting, use a file to file down the sides of the cut-out.
4. Drill a 5/16-inch hole at the center of each of the cut-outs. This will be used to secure the O-clamps to the ABS with a screw.
5. To assemble, place the O-clamps into the cut-out and run a 3/8-inch x 2.5-inch-long screw through the hole in the O-clamp and the hole in the other side of the ABS pipe. Similarly secure the enclosure by feeding two 3/8-inch x 2.5-inch-long screws through the base of the enclosure and the holes made in the horizontal piece of ABS. Add locking nuts on the ends of all the screws to secure them. In the case of the enclosure screws, add three washers before the nuts, so fill the gap.

6.3 Communication Array Circuit

6.3.1 Bill of Materials for Electrical Component

Table 8: Bill of Materials for Electrical Components

#	Part/Material	Price ($CAD)/ Unit	Link
1	Micro SD Storage	1.875	microSD Card Storage Link
2	ESP32 WROOM-32E	12.00	ESP32 Link
3	Speaker	5.09	Speaker Link
4	MAX98375	1.93	Amplifier Link
5	Buttons	2.03	Buttons Link
6	Battery Pack	15.99	Battery Link
7	Protoboard	0.40	Protoboard link
8	Lead Solder	0.05	Solder Link
9	USB to USB Micro	11.06	Cable Link
10	Solid Wire	0.05	Wire Link

6.3.2 Equipment list

The equipment used will be identical to that of the BOM with the exception of 2 wood pieces used to mount items. In this subsystem we used: Micro SD Storage, ESP32 WROOM-32E, speaker, MAX98375 Amplifier, Buttons, Battery pack, protoboard, lead solder, USB to USB micro cable and solid wire. The additional pieces used were a thin small wood chamfer used to mount the speaker. Secondly, we added another thin piece of wood under the battery pack to give it elevation in the enclosure.

6.3.3 Instructions

1. Create the circuit by inserting pins and directly soldering them to the proto board using the reference picture shown below.

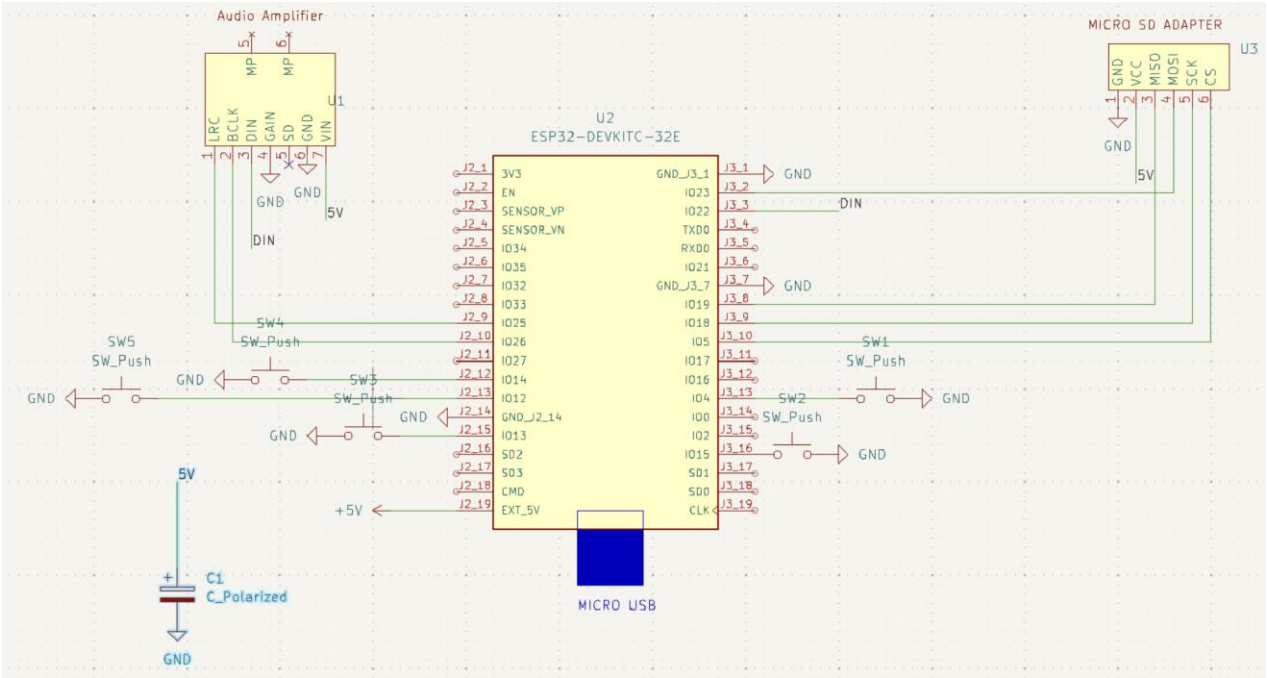

Figure 13: Screenshot of the Protoboard Schematic

2. Use the 4 mounting holes of the protoboard, drill these 4 corners into the enclosure as shown below.

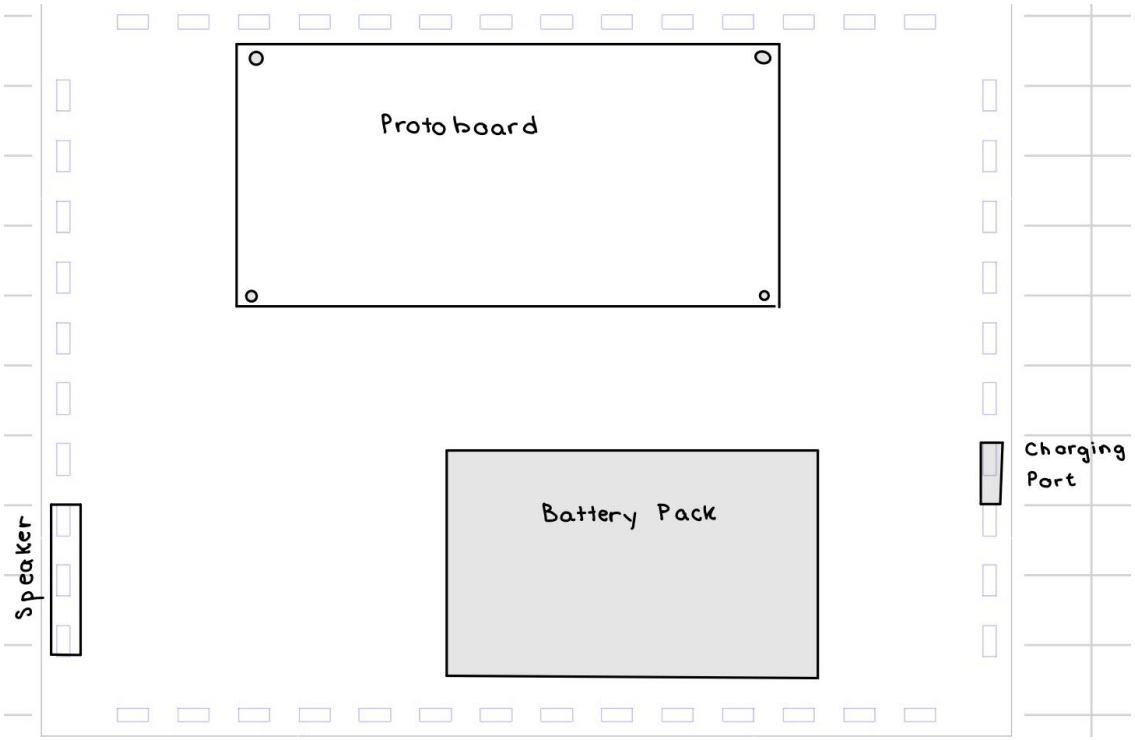

Figure 14: Diagram of the Electrical Component Layout within the Enclosure

3. Epoxy the battery pack to the enclosure itself.
4. Epoxy the speaker to the wooden chamfer.
5. Connect any computer to the ESP32 where the code in the Figure X. will be uploaded.
6. After the uploading process is complete you may disconnect your laptop to the microcontroller.
7. Using the USB to USB-micro cable, connect the ESP32 microcontroller to the battery pack.
8. Ensure that the battery pack is powered on.
9. After the enclosure lid is on, you may solder each button to it's appropriate pin labelled in Figure X.

6.4 Application

6.4.1 Bill of Materials for the Software Component

The application subsystem for the SpeakEasy device comprises several libraries, platforms, and tools necessary for its development and deployment. Below is a comprehensive list of materials used:

1. **Software Libraries and SDKs**:
    - **Android Studio**: Integrated Development Environment (IDE) for creating the application.
      Cost: Free
      Link: Android Studio
    - **Espressif ESP32 Wi-Fi SDK**: Used for communicating with the ESP32 module on the device.
      Cost: Free
      Link: ESP32 SDK
    - **TTS Engine**: Google's Text-to-Speech API for generating MP3 prompts.
      Cost: Free tier available; $4.00 per 1 million characters.
      Link: Google Cloud TTS
2. **Libraries**:
    - **Retrofit**: For managing HTTP communication between the app and the ESP32 Wi-Fi module.
      Cost: Free
      Link: Retrofit
3. **Devices and Hardware**:
    - **Smartphone/Tablet**: Required for testing and using the app.
      Cost: Varies based on the device.
      Example: $150-$1,000.
4. **Hosting Platforms**:
    - **GitHub**: For version control and collaboration.
      Cost: Free (basic plan).
      Link: GitHub

6.4.2 Equipment list

The following equipment was needed during the development and testing of the application subsystem:

- **Development Laptop**: Used to code and test the application in Android Studio.
  Example: MacBook Air, $999.00.

- **Wi-Fi Router**: For testing connectivity between the app and the SpeakEasy device.
  Example: TP-Link Archer A7, $59.99.

- **ESP32 Microcontroller with Wi-Fi Module**: Required for real-world connectivity tests.
  Example: ESP32-WROOM-32, $6.50.

- **Android Device**: For testing the application on various screen sizes and resolutions.
  Example: Google Pixel 4a, $349.00.

- **Power Supply for ESP32**: A USB cable and adapter to keep the ESP32 powered during testing.
  Cost: $10.00.

6.4.3 Instructions

Step-by-Step Guide to Building the Application Subsystem:

Step 1: Setting Up the Development Environment

1. Install Android Studio on your computer by downloading it from the official site.
2. Configure the Android Virtual Device (AVD) to emulate different Android versions for testing.
3. Clone the project repository from GitHub into Android Studio.
4. Sync the project with Gradle files to ensure all dependencies are installed.

Step 2: Designing the Interface

1. Open the res/layout folder in Android Studio.
2. Design the app's home screen with a grid layout for the four prompt buttons.
3. Add a navigation bar at the bottom for easy access to settings and help.
4. Use color-coded placeholders for the buttons to correspond with the physical device.

Step 3: Implementing Functionality

1. Use Retrofit to create an API service for communicating with the ESP32.
2. Integrate Google Cloud TTS API by adding the API key to the strings.xml file.
3. Write functions to:
    - Input text for prompts.
    - Generate and preview TTS audio files.
    - Upload audio files to the ESP32 microSD card.
4. Test each function independently using log outputs and debug tools in Android Studio.

Step 4: Connecting to the Device

1. Write a Wi-Fi connection handler using ESP32 credentials.
2. Test the connection stability by sending dummy data and verifying the response.

Step 5: Testing and Debugging

1. Test the app on both emulated devices and real Android phones.
2. Use network analyzers to ensure the Wi-Fi communication is seamless.
3. Debug any crashes using Android Studio's logcat tool.

Step 6: Deployment

1. Generate an APK file for distribution.
2. Test the installation on multiple Android devices for compatibility.

Step 7: User Interface Enhancements

1. Add customization options for button colors and prompt labels.
2. Implement real-time feedback for caregivers, such as displaying active prompts or battery status.

6.5 Testing & Validation

6.5.1 Electrical Testing

The first test is the functionality and loudness of the speaker. Reflecting on our previously created assumptions, we noted that the speaker must work in the circuit as well as be loud enough to alert his caregivers. This specific prototype focuses strongly on our design for reliability and usability. This prototype will be accomplished by configuring a new circuit that is capable of reading the microSD card inserted in the microSD card storage as well as playing the specific mp3 documents stored in those files. In addition, the circuit must work reliably which means it behaves consistently in a manner of good quality or performance.

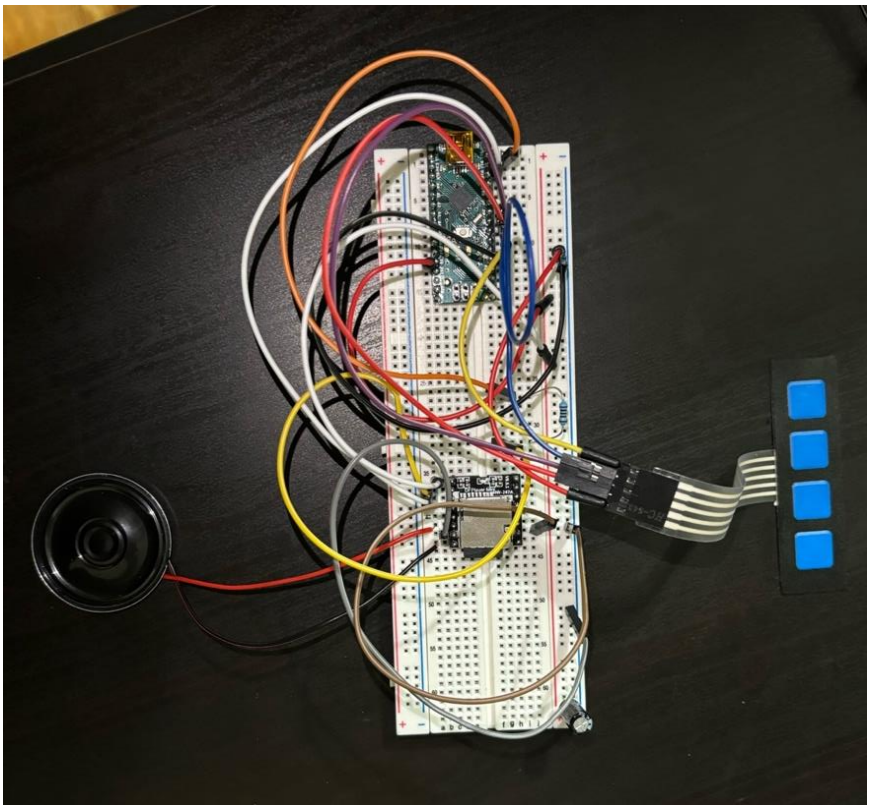

Figure 151: Electrical Breadboard Platform (Top View)

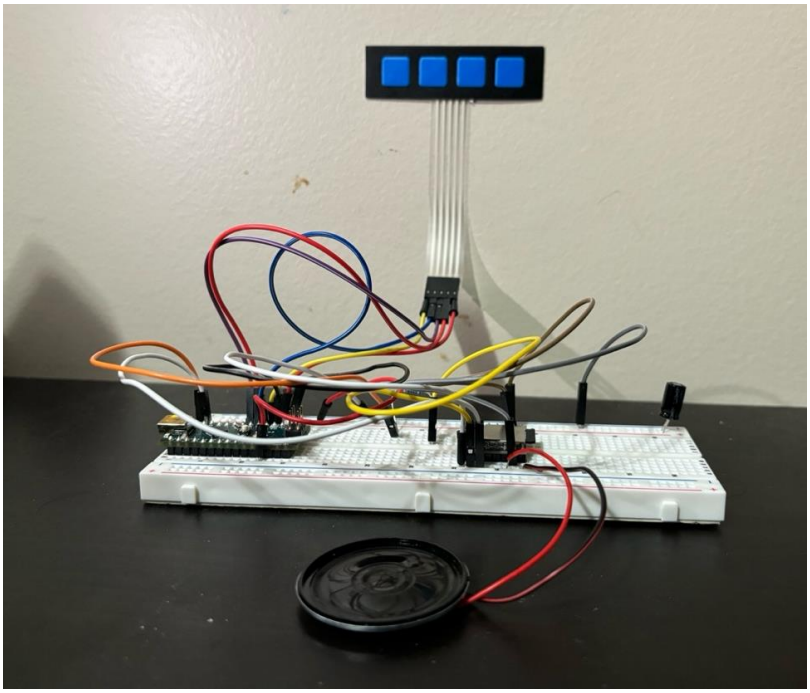

Figure 162: Electrical Breadboard Platform (Side-View)

After completing the appropriate tests, we achieved a few metrics that accurately solved our prototype tests. Our original ideal value was for the speaker to be greater than or equal to 80dB (decibels). Our accepted marginal value was between 70 - 100 decibels. After have completing a variety of tests we've been able to figure out the maximum, minimum and appropriate decibel levels.

When playing mp3 files through the DF Player Mini, we have the option of selecting at what volume each file plays at (between 0 and 30). Our first test was to measure the maximum decibels that our 2W, 8 Ohm speaker could produce. After testing various prompts while setting the volume at 30, we achieved a maximum of 115dB. This was achieved through using multiple websites, different settings as well as using both a phone and laptop to measure the total sound. The websites used are: Youlean, CheckHearing and Enviraj Consulting. Multiple websites were used to verify that the correct sound level was being read. Different devices were used to ensure that the device itself didn't play an impact. As a result, both devices being run at the same time had no effect and obtained the same readings. Lastly, this test was conducted both in my bedroom (silent environment) and in the SITE building on the University of Ottawa campus during a career fair (extremely loud and busy environment). This again showed no impact and read the same levels due to the sound emitted from the speaker being louder than the surrounding sound.

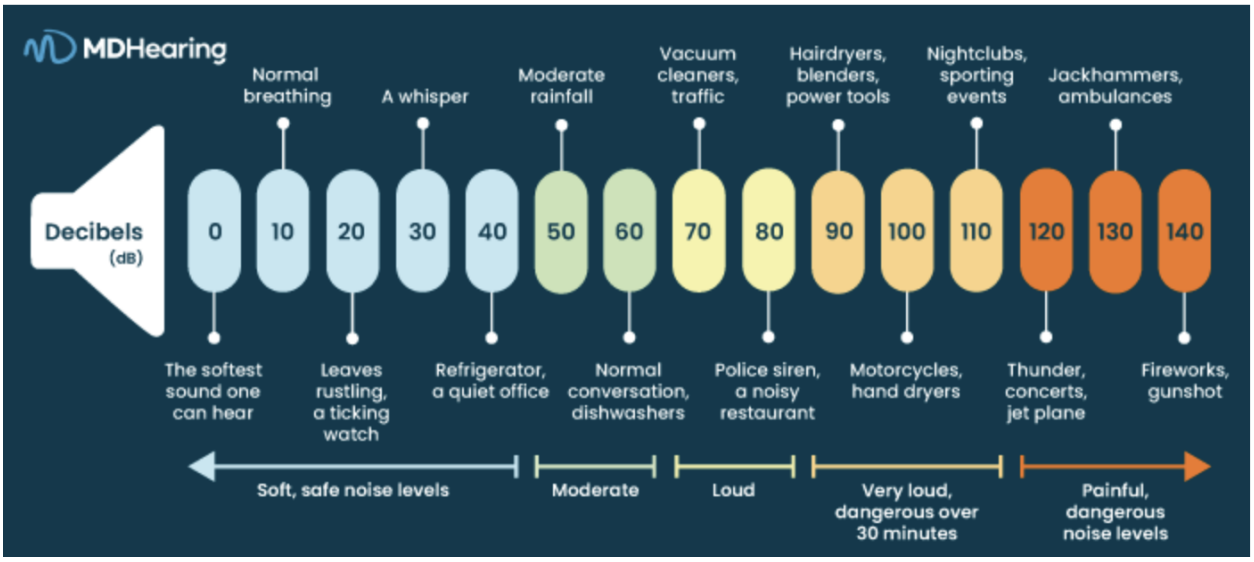

Figure 173: Decibel chart acquired from MDHearing (Decibel Chart: All You Need to Know, n.d.)

The optimal value needed to be used in our final model will be in the range between 85 to 100 dB. This will vary depending on the prompts chosen. This value was determined by experimenting with what's the maximum distance possible before the speech become inaudible. After conducting multiple tests in both environments listed above, we achieved a maximum of 44 feet (13.4 meters) of distance. This is sufficient for the environment that our user is in given that the room itself was much quieter and was contained in a room of similar length.

Another prototype made was to test the functionality/actuation of the input buttons, as well as how they mount and stay into a given hole. First, we tested exactly when the button actuated during a button press. It was found that the button consistently actuates slightly before the tactile "click" sound is heard. This was great, as the consistency of actuation was unhindered by where

on the button you clicked or what angle you click the button at, as the “click” noise would always signal a button press.

Also, this prototype tested the strength and durability of the button mounting system, being two one-way bendable prongs. There is no adhesive of any kind holding the button to the plate, only the prongs within the button design. Even so, the button was firmly in place and would only move if pressed upon from the inside of the enclosure – something that would never happen with ordinary use. Below is the prototype used for these two tests.

Figure 18: Button Testing Prototype

6.5.2 Mechanical Testing

Mechanical Prototype Test 1

The purpose of the first test was to estimate the performance of the interface between the truss tube clamps and the PVC tubing by testing the resistance they have against failure caused by weight applied to the shelf supporting the enclosure.

To perform this test, a 3D print was made to mimic the dimensions of the truss clamp and was fixed to a machined piece of PVC. The 3D print had one end with the same shape and hole as the rectangular portion of the truss clamp which houses the screw that will interface with the PVC tube. Material was added to this geometry to have a hole for hanging weight. After the part was 3D printed, threads were added to the screw hole. This 3D print will be referred to as the clamp in this test, as it mimics the role of the clamp in the clamp-PVC interface.

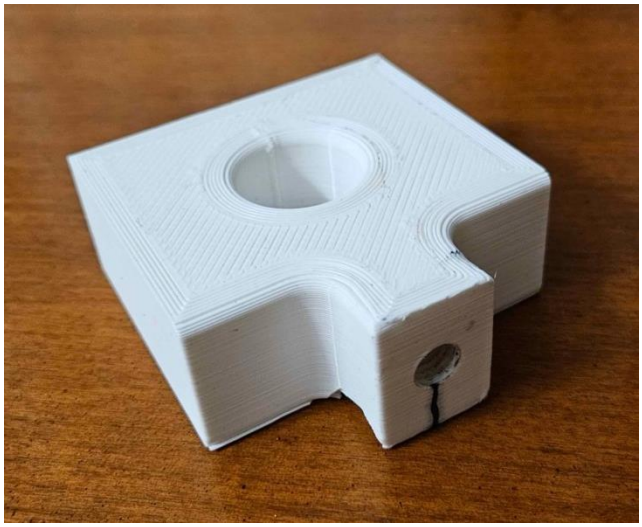

Figure 19: PVC Mount Prototype

To prepare the PVC tube, a rectangular shape the size of the truss clamp edge was carved out of the PVC for the 3D printed to be inserted. Following this, a hand drill was used to create a hole on the PVC tube across from the rectangular cut-out. The 3D printed piece was then inserted into the rectangular cut-out of the PVC tube and a screw was threaded through the 3D print and PVC tube.

To test the resistance of the mechanism to weight applied on the shelf, a bucket was attached to the hanging hole of the 3D print using zip-ties, and the clamp-PVC joint was held in the orientation that it will have when mounted to the wheelchair. Weight was then progressively added to the bucket using increments of water volume and eventually a steel cylinder to determine the maximum load that this joint could withstand before failure.

Figure 20: PVC Load Testing

Table 9: Test for Reliability

Test	# of water bottles	Total weight on joint (kg)	Duration (seconds)	Result
1	0	0.9	10	Good condition
2	2	2.0	10	Good condition
3	6	4.1	10	Good condition
4	10	6.2	10	Good condition
5	14	8.3	10	Good condition
6	20	11.5	10	Good condition
7	26	14.7	10	Good condition
8	32	17.0	10	Good condition
9	36	20.0	10	Good condition
10	36	20.0	46	Good condition
11	6x6x6" cylinder of steel	27.7* *Assuming high carbon steel	600	Deformation of joint, no failure.

We anticipate that the communication device and fixtures to create its shelf will not exceed 4 kg. As a result, this test reveals that the proposed joint between the PVC tube and the clamp can resist a prolonged load that is three times the maximum expected weight. Additionally, this resistance will be improved with the intended use of a second clamp that will interface with the wheelchair frame and the PVC pipe.

Mechanical Prototype Test 2

The purpose of the second mechanical test was to determine the strength of the proposed shelf. This prototype was created using ABS tubing and inserting a 90-degree PVC elbow and a PVC Tee to create the arm which mounts to the wheelchair and the shelf to support the communication array enclosure. The resistance to bending was tested by fastening a bucket to the distal extremity of the shelf and adding aluminum cylinders to the bucket which amounted to a weight of 25 lbs. The shelf was subject to this weight for 15 minutes and the deflection was observed using a level.

Figure 21: Plastic Shelf Bending Test Set-Up

The plastic shelf was level when the bucket was fastened to the Tee coupling prior to the test. Once the aluminum cylinders were added to the bucket, the shelf was visibly deformed and was no longer level. At the end of the 15 minutes with the 25lb load, the shelf became slightly less deformed but did not return to its initial level state. The load used in this test greatly exceeds the anticipated load on the communication array and its shelf. As a precaution, a bracket should be added to the 90-degree bend of the structure to avoid any plastic deformation during the use of the device.

Figure 22: Level of Shelf Before, During, and After Bending Test (Left to Right)

6.5.3 Software Testing

Part 1 | Arduino Software Testing

1. **Section Test**:
    - **Objective**: Verify that the Arduino correctly detects button presses and associates them with the appropriate prompts.
    - **Method**: Connect the Arduino to a serial monitor and press each button 50 times. Check if the correct prompt ID is printed for every press.
    - **Success Criteria**: 100% accuracy in detecting button presses without duplicate or missed inputs.

2. **Audio Playback Test**:
    - **Objective**: Confirm that the correct audio file is played when a button is pressed.
    - **Method**: Upload five test MP3 files to the microSD card, each corresponding to a button. Press each button and record the audio output using a sound meter.
    - **Metrics**: Ensure no latency greater than 200 milliseconds between the button press and the audio output.
    - **Validation**: Audio clarity must achieve a minimum score of 85% intelligibility in a noise test (e.g., 50 dB ambient noise).

3. **MicroSD File Management Test**:

- **Objective**: Validate the Arduino's ability to handle file storage, overwriting, and retrieval on the microSD card.
- **Method**: Perform read/write tests by uploading and replacing files of varying sizes (100 KB to 500 KB).
- **Success Criteria**: Zero read/write errors and proper file overwriting functionality.

4. **Wi-Fi Communication Test**:
    - **Objective**: Ensure reliable connectivity between the Arduino and the mobile app.
    - **Method**: Establish a connection between the app and the Arduino 30 times under various conditions (distance, interference).
    - **Metrics**: Connection success rate of at least 95% within a range of 5 meters.

5. **Stress Test**:
    - **Objective**: Validate the Arduino system's stability under continuous operation.
    - **Method**: Simulate 1,000 button presses in a controlled environment and measure system uptime.
    - **Success Criteria**: No crashes, resets, or performance degradation during the test.

Part 2 | Arduino Software Testing

1. **User Interface Usability Test**:
- **Objective**: Evaluate the app's ease of use and responsiveness.
- **Method**: Conduct a user test with 10 participants (5 caregivers and 5 non-technical users). Ask them to complete tasks such as modifying a prompt, previewing audio, and uploading it to the device.
- **Metrics**: Average task completion time should not exceed 30 seconds, and 90% of participants should rate the app at least 8/10 for ease of use.

2. **Text-to-Speech (TTS) Functionality Test**:
- **Objective**: Confirm that the app accurately converts text input into high-quality MP3 files.
- **Method**: Input 20 diverse text strings into the app and analyze the generated MP3 files for pronunciation accuracy and audio quality.
- **Validation**: Accuracy score of 95% or higher in pronunciation, and a minimum bit rate of 128 kbps for audio files.

3. **File Upload Validation**:

- **Objective**: Test the process of transferring MP3 files from the app to the Arduino's microSD card.
- **Method**: Upload 10 files of varying sizes (100 KB to 500 KB) and verify their presence and functionality on the Arduino.
- **Metrics**: Upload success rate of 100% with file transfer times not exceeding 5 seconds per file.

4. **Connectivity Stability Test**:

- **Objective**: Ensure stable and secure communication between the app and the Arduino over Wi-Fi.
- **Method**: Test the connection across different devices (Android phones and tablets) and Wi-Fi environments.
- **Metrics**: Connection success rate of at least 98% within a 5-meter range.

5. **Error Handling Test**:

- **Objective**: Validate the app's ability to handle errors gracefully (e.g., failed uploads, connection loss).
- **Method**: Simulate errors such as network disconnection during file upload and check for appropriate error messages.
- **Success Criteria**: All errors should prompt clear and actionable feedback for the user.

6. **Stress Test**:

- **Objective**: Test the app's stability during continuous use.
- **Method**: Run the app for 1 hour while performing periodic uploads, edits, and connections to the Arduino.
- **Success Criteria**: No crashes, freezes, or performance degradation.

7 Conclusions and Recommendations for Future Work

In conclusion, the biggest lesson learned was to always reach out to those with experience whenever appropriate. Whether it's asking a question, asking for feedback or suggestions on prototypes, or even just asking for wisdom on a topic for the situation you find yourself in, there

will always be valuable information for you to gather. Future work on the project includes adding additional features that are standard to AAC's, such as a power switch, as well as our own ideas, like a liquid crystal display (LCD) to visually display the chosen prompt. This would be in case the caretakers don't understand or are not in the vicinity of the user. Additionally, upgrading the enclosure's material to a stronger textile that is waterproof would significantly improve its quality. A material such as plastic or acrylic would be ideal for this purpose.

APPENDICES

8 APPENDIX I: Design Files

Table 10:3 Referenced Documents

Document Name	Document Location and/or URL	Issuance Date
Enclosure_LaserCut_V6.pdf	https://uottawamy.sharepoint.com/personal/erudd053_uottawa_ca/_layouts/15/guestaccess.aspxshare=ErE59nTQHiREoXPUJN_oVR8BUH_G59cuxwLuq0Osi0O17A&e=tPswr9 Also available in the COM1 Makerrepo repository	3/12/2024
Enclosure_LaserCut_V6.svg	https://uottawa-my.sharepoint.com/personal/erudd053_uottawa_ca/_layouts/15/guestaccess.aspx?share=EWyf7bfUQmhNkxcqcXWQO3cBbayfJuBtuNTpKSTxAqKwmw&e=valXtn	3/12/2024
FV_P6 - EdgeGuard(square).gcode	https://uottawa-my.sharepoint.com/personal/erudd053_uottawa_ca/_layouts/15/guestaccess.aspx?share=EXuaDpxVmj1LrF1DBQ5u0RwBXGJJlgjbQ4t-mkfzIsHMrg&e=ra8x6R	3/12/2024

9 APPENDIX II: Other Appendices

The initial svg file for the enclosure was generated using boxes.py. Specifically, the ”BasedBox” template was used with the following specifications:

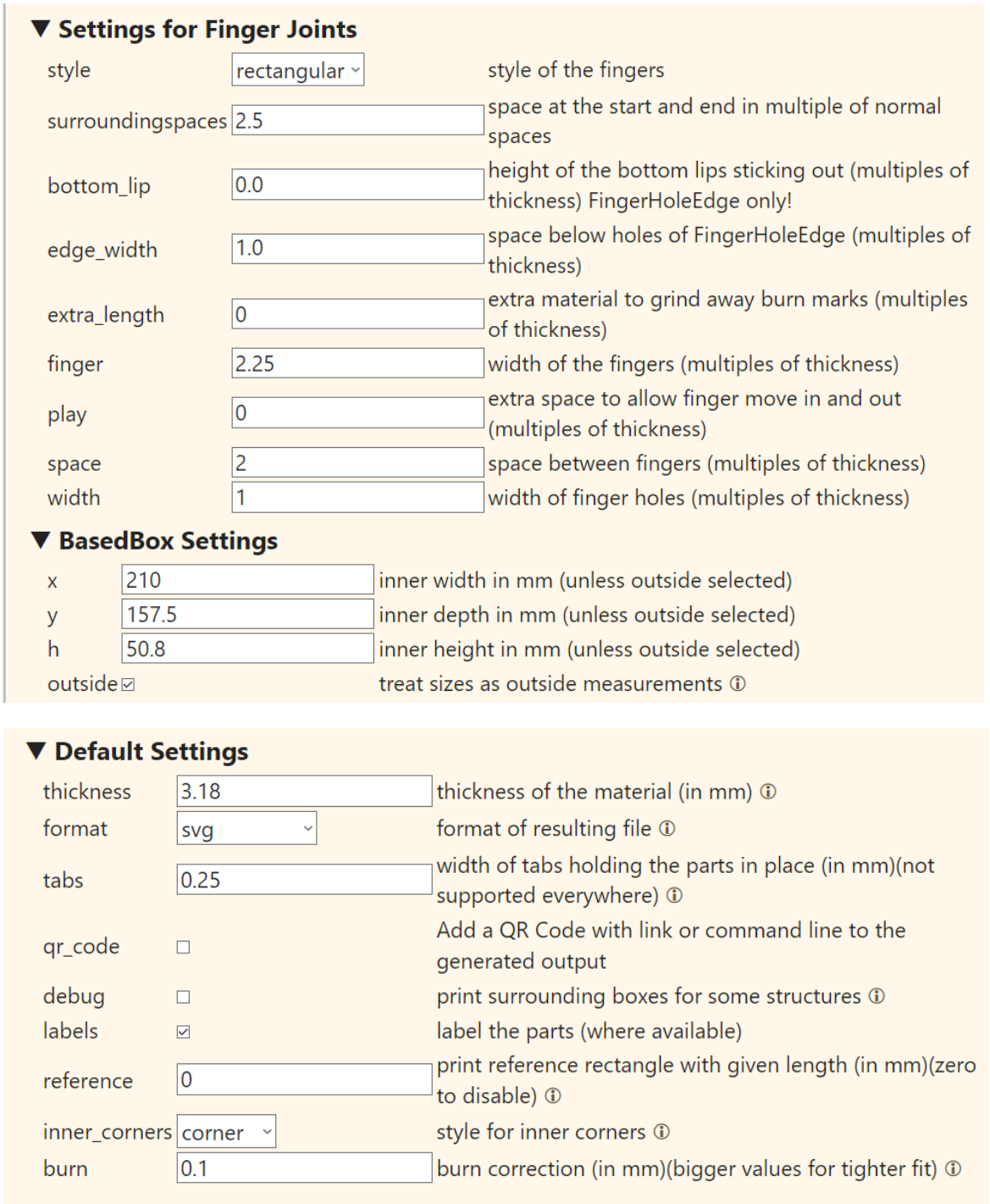

Figure 23: Screenshot of BasedBox Template Values Used