

ECE 445 - SENIOR DESIGN LABORATORY  
DESIGN DOCUMENT  
03.03.2025

# Audio Augmented Reality Glasses

## **AARG**

Team #14

Nikita Vasilyev (nvasi2)

Sunny Chen (sunnyc3)

Evan Chong (eschong2)

TA: Aishee Mondal

# Table of Contents

<b>Table of Contents</b>	<b>1</b>
<b>1 Introduction</b>	<b>3</b>
1.1 Problem	3
1.2 Solution	3
1.3 Visual Aid	4
1.4 High-Level Requirements	4
<b>2 Design</b>	<b>5</b>
2.1 Block Diagram	5
2.2 Physical Design	6
2.3 Physical Subsystem	9
2.3.1 Design Overview	9
2.3.2 Functionality and Contribution	10
2.3.3 Requirements and Verification	10
2.3.4 Design Decisions	11
2.4 Peripheral Subsystem	12
2.4.1 Design Overview	12
2.4.2 Functionality and Contribution	13
2.4.3 Requirements and Verification	14
2.4.4 Design Decisions	15
2.5 Power Subsystem	16
2.5.1 Design Overview	16
2.5.2 Functionality and Contribution	17
2.5.3 Requirements and Verification	20
2.5.4 Design Decisions	21
2.6 Communication Subsystem	22
2.6.1 Design Overview	22
2.6.2 Functionality and Contribution	22
2.6.3 Requirements and Verification	23
2.6.4 Design Decisions	24
2.7 Application Subsystem	25
2.7.1 Design Overview	25
2.7.2 Functionality and Contribution	25
2.7.3 Requirements and Verification	25
2.7.4 Design Decisions	26

	2
2.8 Tolerance Analysis	27
2.8.1 Application Subsystem - Classification Model Selection	27
2.8.2 Power Subsystem - LDO vs. Buck Converter	28
<b>3 Cost and Schedule</b>	<b>31</b>
3.1 Cost	31
3.1.1 Labor	31
3.1.2 Parts	31
3.1.3 Total Cost	32
3.2 Schedule	33
<b>4 Ethics and Safety</b>	<b>35</b>
4.1 Ethics	35
4.2 Safety	35
<b>5 References</b>	<b>36</b>
5.1 Datasheets	36
5.2 Ethics/Safety	37

# 1 Introduction

## 1.1 Problem

Have you ever seen an object that piqued your interest, but you didn't have an efficient way of researching what it was? Repeatedly searching online to identify the subject can be a lengthy and tedious task, and this is the problem we seek to address. Our solution is meant to enlighten our users about unknown plants, animals, or objects in any setting they are observing.

Furthermore, an auditory aid can be further applied to helping the visually impaired, giving context to the surroundings that they would otherwise be unable to fathom or see. A multifaceted solution to spatial object recognition problems is what we are trying to accomplish with the AARG glasses.

## 1.2 Solution

Our project idea stems from the surge of AR prototype glasses being introduced over the past year. We are planning to create our own glasses, but in contrast to those on the market, ours will focus on the audio experience of the user. These glasses will have the explicit capability of capturing images of objects and relaying this information to an application that will process these images in the backend. The application will then send an explanation of the object back to an audio device on the glasses (either a speaker or a bone-conducting device).

The glasses will essentially work as a digital tour guide, with the explanation of the object being auditory rather than visual. The use case we have decided to tackle is a botanical tour guide, but the purpose is to create a platform that other applications can utilize for their objectives. Other objectives include but are not limited to the following:

- Accessibility for the visually impaired
- Language translation
- Tourism and museum tour guide
- Security and Surveillance

## 1.3 Visual Aid

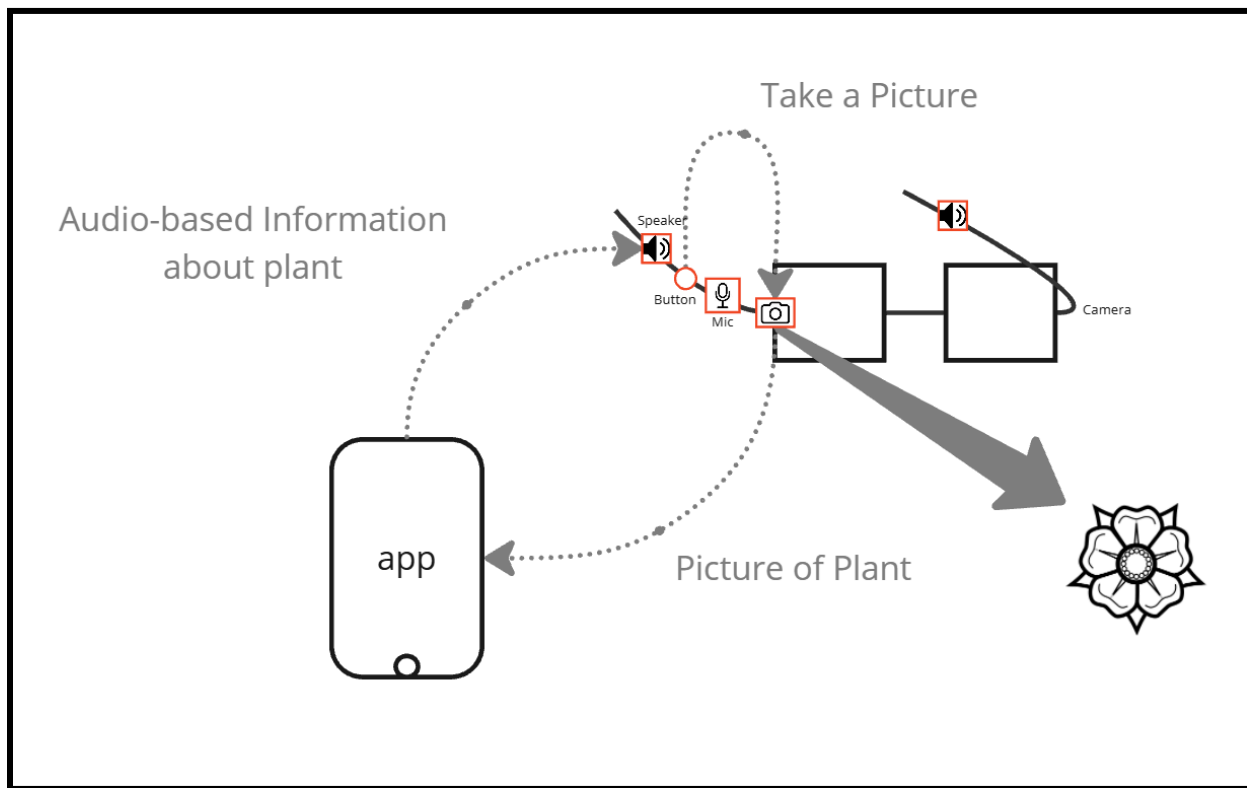


Figure 1: Visual Depiction of Example User Flow

## 1.4 High-Level Requirements

- **Fast Computation Time**
  - The time taken for the user to obtain audio feedback from the glasses system should not exceed 12 seconds. This value was obtained from experimentation of timing LLMs trying to identify unknown plants, along with references based on the summation of GPT backend processing time with hardware communication relay times.
  - Reach Goal: 6 seconds
- **Accuracy**
  - The final prototype should be able to correctly identify plants 85% of the time. This will be based on our application subsystem, which will use a model to determine the plant's classification. We will test the device on 20 plants and expect to get 17 out of the 20 correct. In the case of poor camera framing, we will allow for each plant to be captured up to 3 times.

- Reach Goal: 90% accuracy
- **Final Prototype Weight**
  - The final prototype will weigh no more than 200 grams. We obtained this value from researching similar augmented reality headsets and aim to make our final prototype lighter than headsets on the market but still lightweight enough to be comfortably worn similar to commercial glasses, throughout the day.
  - Reach Goal: 100 grams

## 2 Design

### 2.1 Block Diagram

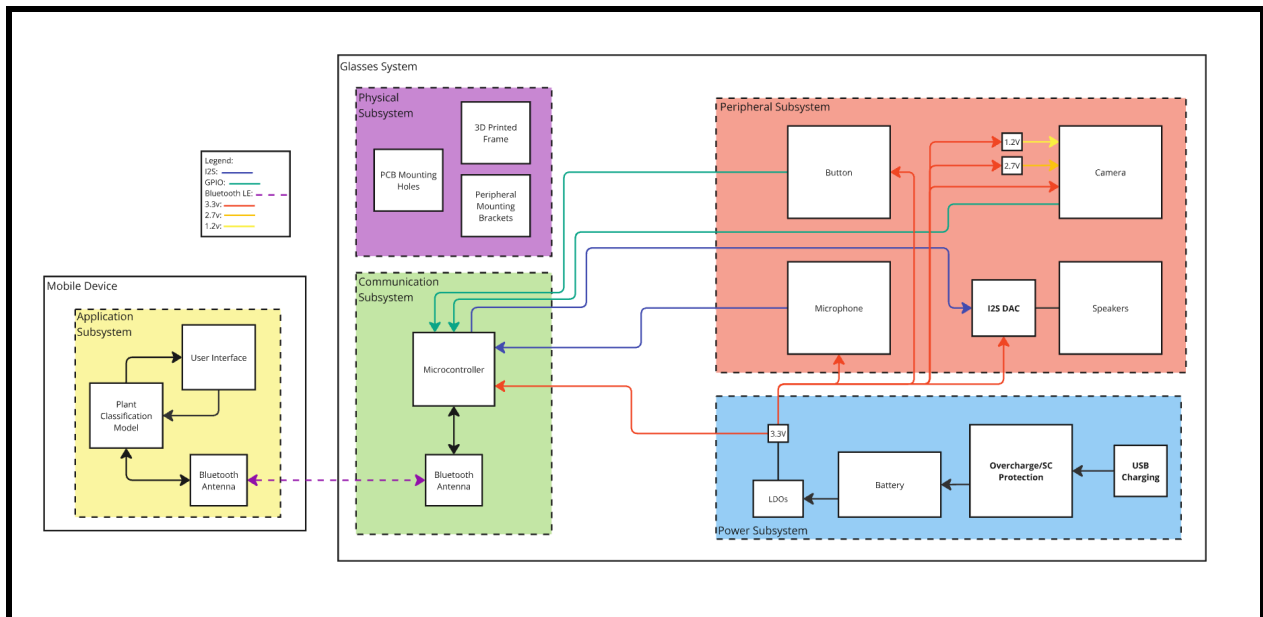
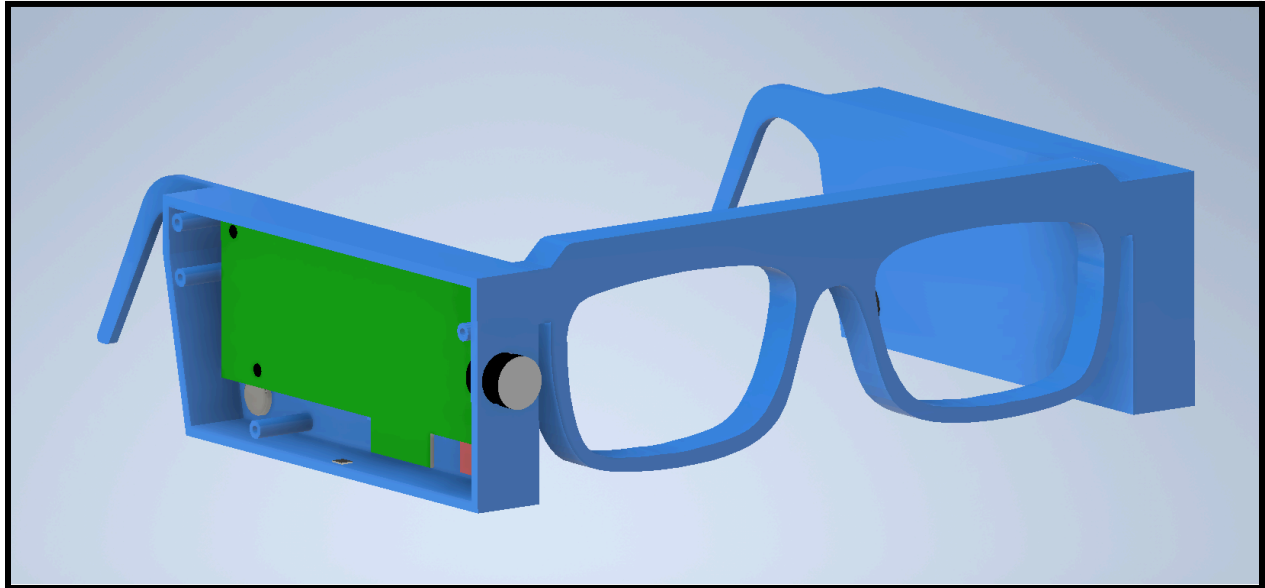


Figure 2: Block Diagram of AARG Subsystems

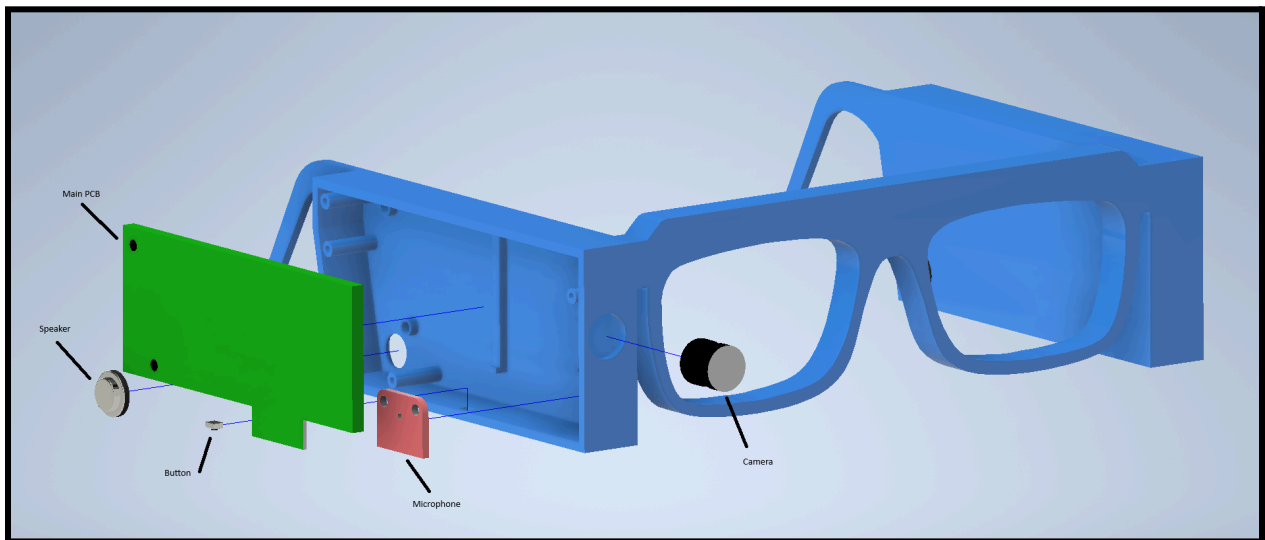
This block diagram depicts the subsystems present within the glasses and mobile device. The glasses platform contains the physical, peripheral, communication, and power subsystems. The mobile device platform contains the application subsystem. The peripheral subsystem consists of all devices that interface with the user and the outside world (speaker, microphone, button, camera). The power subsystem is used to power all the sensors and devices onboard the glasses platform. The communication subsystem uses the ESP32 and Bluetooth antenna to facilitate communication between the mobile device and the peripherals. The physical subsystem consists of mounting holes and brackets for all devices alongside the

frame in which they will all fit. Within the mobile device, the application subsystem is tasked with using the peripherals to complete an app-driven task.

## 2.2 Physical Design



*Figure 3: 3D Model of AARG's Right Side View*



*Figure 4: Labeled Exploded View of AARG's Right Side*

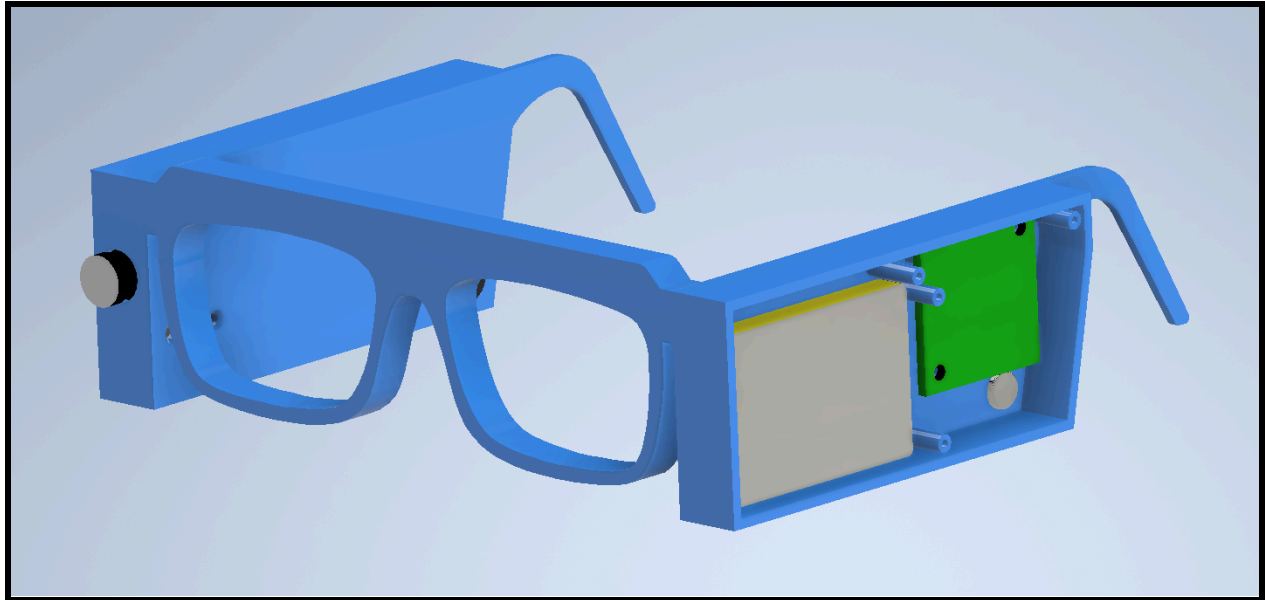


Figure 5: 3D Model of AARG's Left Side View

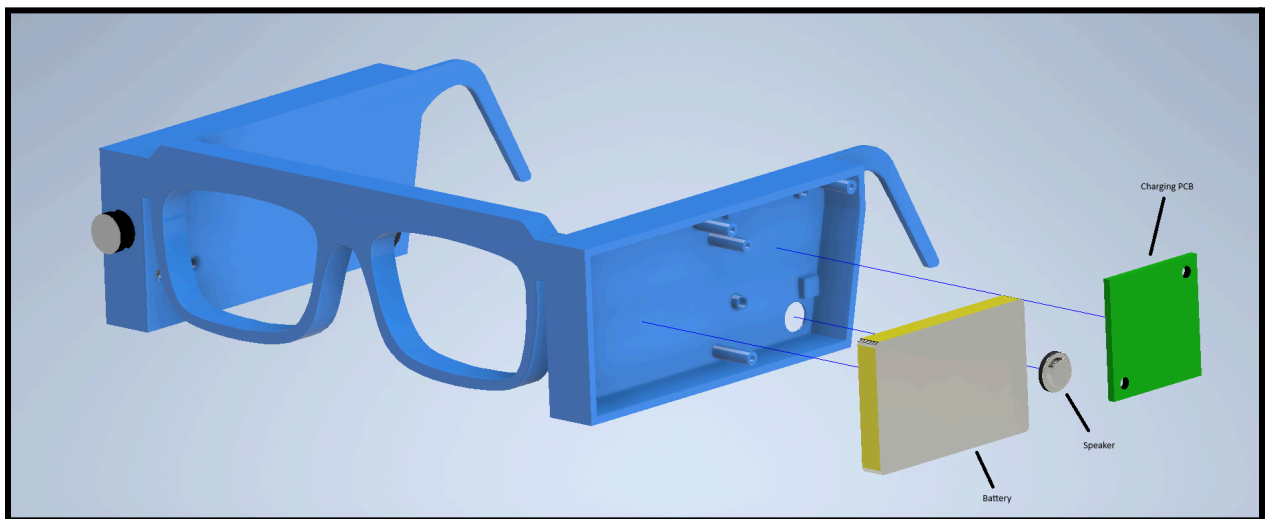


Figure 6: Labeled Exploded View of AARG's Left Side

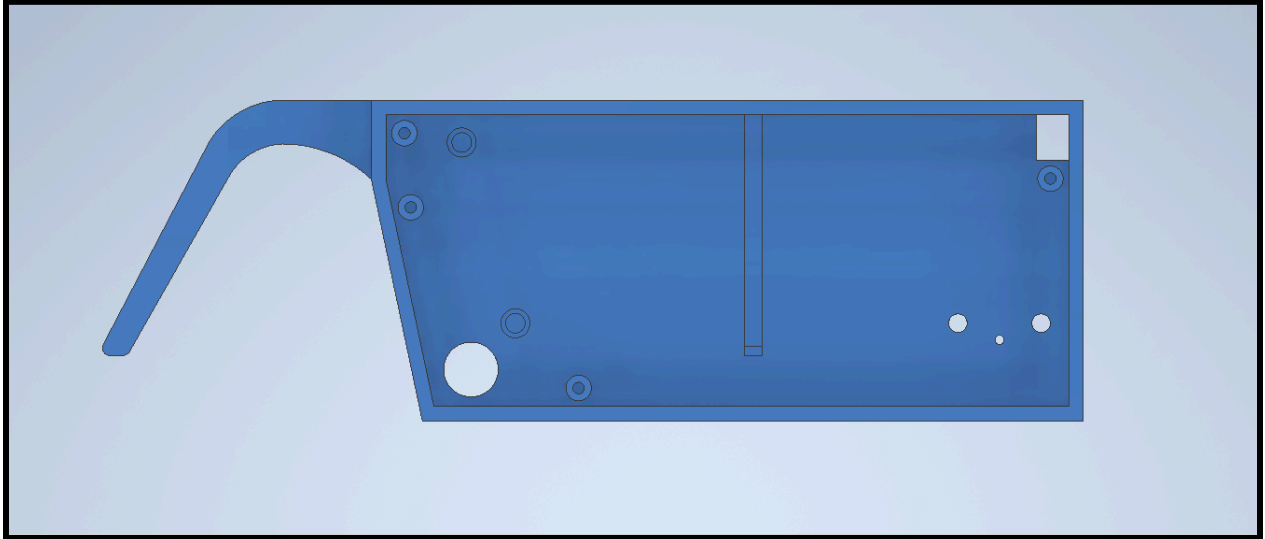
The physical design of the AARG is focused on resembling the form of typical glasses with additional compartments on the left and right arms to house/mount the required components. The right side will comprise the main PCB, right speaker, camera, microphone, and interact button. The left side will contain the battery, left speaker, and battery charging PCB. The goal of this breakdown is to do the best job possible while balancing the weight and minimizing the number of wires required to come across the frame. The frame will be 3D printed using FDM with PLA, with the potential to use SLA if the accuracy of FDM does not suffice. The frame contains pilot holes for the PCBs and enclosure covers for simple assembly



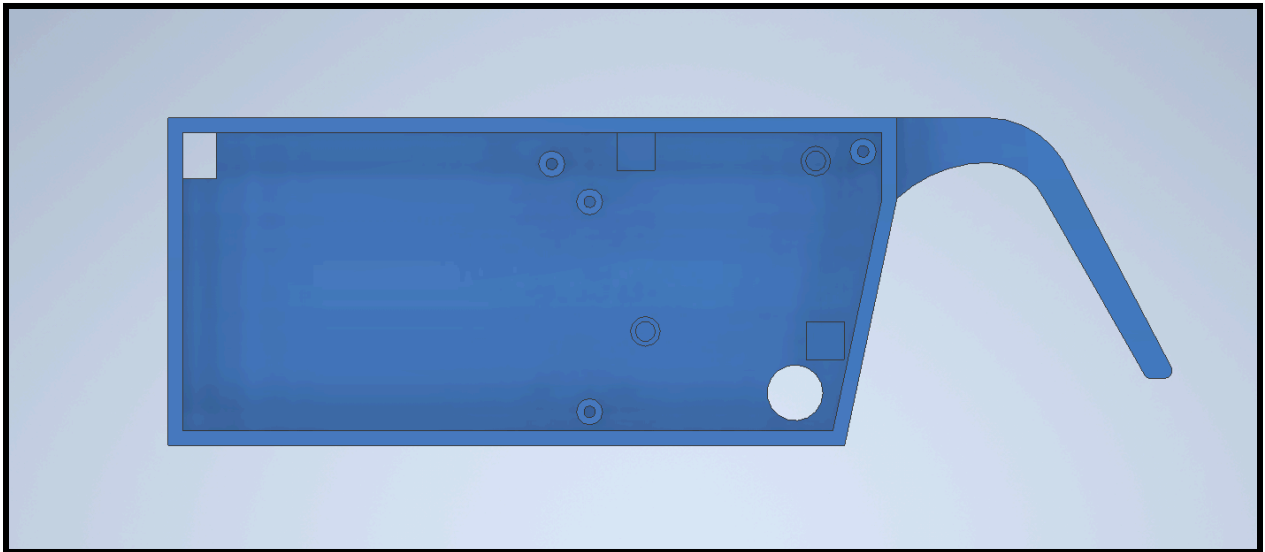
and securing of components. During the final assembly, the enclosure covers will be installed alongside the pilot holes with a liquid O-ring to facilitate water resistance and adhesion.

## 2.3 Physical Subsystem

### 2.3.1 Design Overview



*Figure 7: 3D Model of Right Arm*



*Figure 8: 3D Model of Left Arm*

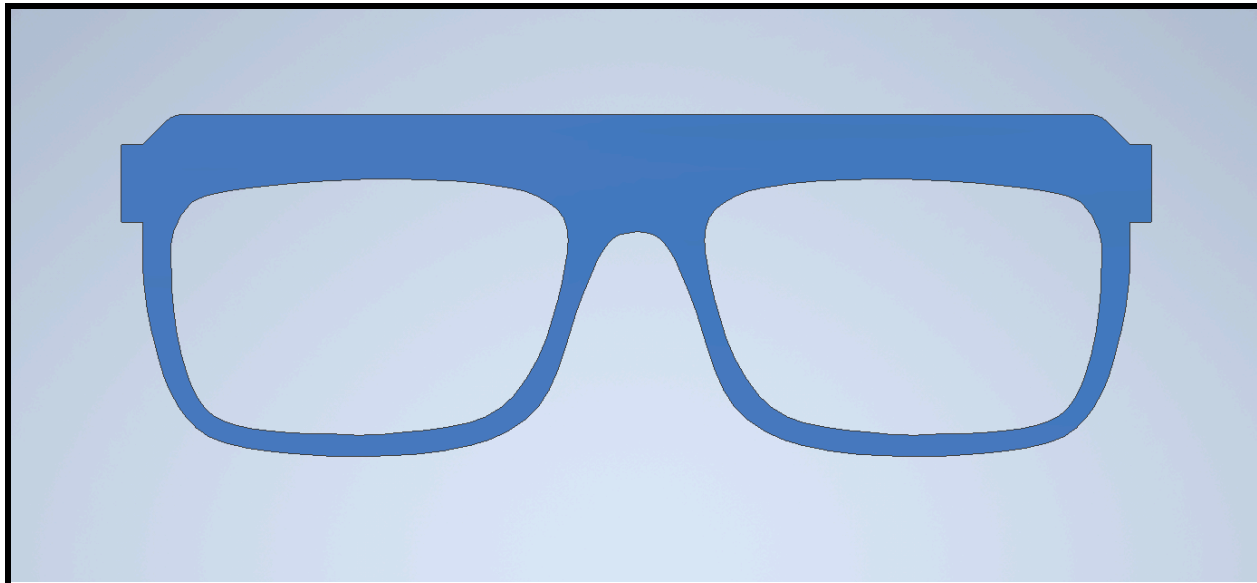


Figure 9: 3D Model of Frame

### 2.3.2 Functionality and Contribution

The purpose of the physical subsystem is to create an enclosure for all electronic components such that they can fit into a pair of glasses. This will be done by designing and 3D printing a set of glasses with hollowed-out arms and custom dimensions to accommodate the PCBs, peripheral, and battery clearances. The mounting of the components will be facilitated by preplanned pilot holes for each part. This subsystem largely contributes to the high-level requirement of wearable total weight. This is done by eliminating unnecessary mounting mechanisms and using lightweight plastic to save weight within the enclosure. Additionally, a significant goal of this subsystem is to ensure the wearability of these glasses. The wearability requirement consists of the glasses not containing any abrasive edges that could scratch the user during wear, having the correct dimensions according to the measurements of the average ears size, and the size of the arms/frame.

### 2.3.3 Requirements and Verification

Requirements	Verification
Components secured firmly in place during normal user operations. There must be no internal movement of components when shaken at a rate of $\frac{1}{2}$ meter per second.	<ul style="list-style-type: none"> <li>• Measure out a distance of <math>\frac{1}{2}</math> meters.</li> <li>• Set up a timer to count down from 10 seconds.</li> <li>• Start the timer and move the glasses in the air left to right, aiming to reach each side at 1-second intervals.</li> </ul>

	<p>During this time, count the number of lengths traveled to verify the final rate.</p> <ul style="list-style-type: none"> <li>● Once the timer expires, halt the process and ensure that the number of lengths traveled is 10.</li> <li>● The test is successful if there is no audible rattling internal of the glasses throughout the test.</li> </ul>
<p>The surface of the glasses does not contain any abrasive textures/sharp edges. The user must not be able to receive any cuts or scratches during normal operation.</p>	<ul style="list-style-type: none"> <li>● Set the device on a flat table.</li> <li>● Using a paper towel, brush up against the outer surface of the glasses. Be sure to brush every surface, including the frame and arms.</li> <li>● Once all surfaces have been brushed two times, inspect the glasses for any signs of debris from the paper towel.</li> <li>● If there is any sign of paper towel on the glasses, the test has failed.</li> </ul>
<p>The final prototype will weigh no more than 200 grams with all components inside.</p>	<ul style="list-style-type: none"> <li>● Put a scale on a flat table and zero out the scale.</li> <li>● Place the glasses on the scale and inspect the shown weight.</li> <li>● If the shown weight is above 200g, the test has failed.</li> </ul>

### 2.3.4 Design Decisions

We decided to 3D print the physical subsystem to ensure ease of adjustments between iterations and easily reproducible models if one were to break during testing. Additionally, 3D printing the model allows us to experiment with different compositions of plastic and printing methods. Another decision that was made about the physical subsystem was the mounting mechanisms. We decided on using pilot holes for our PCB mounting points as opposed to adding threading since it would provide a more secure fit without worrying about the complexity and structural integrity of the threads.

## 2.4 Peripheral Subsystem

### 2.4.1 Design Overview

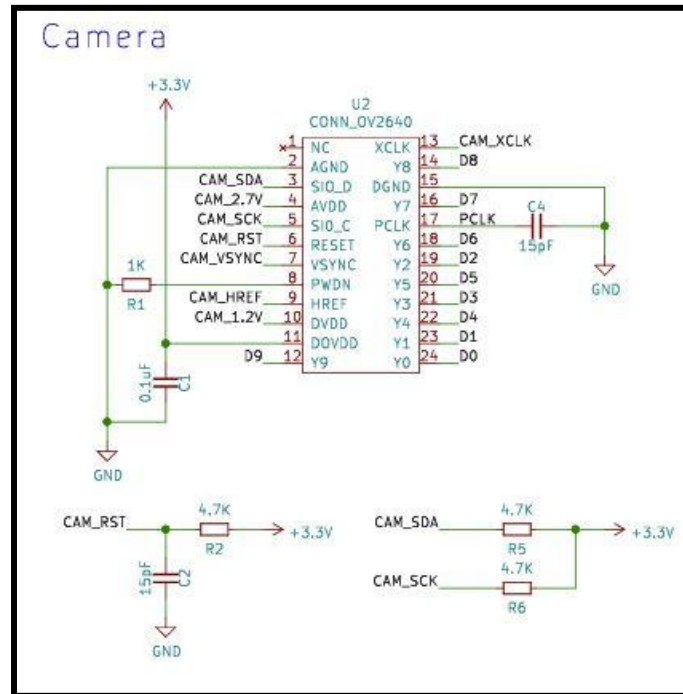


Figure 10: Camera Schematic

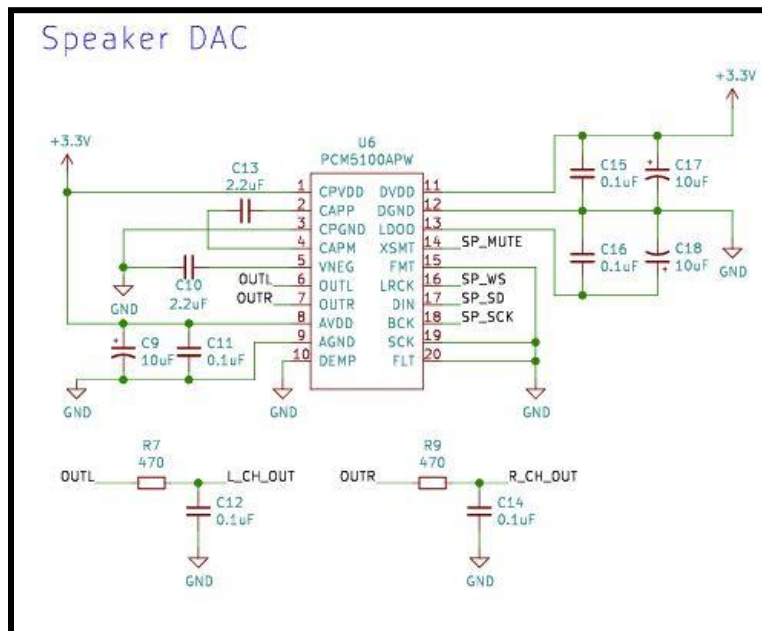


Figure 11: Speaker and DAC Schematic

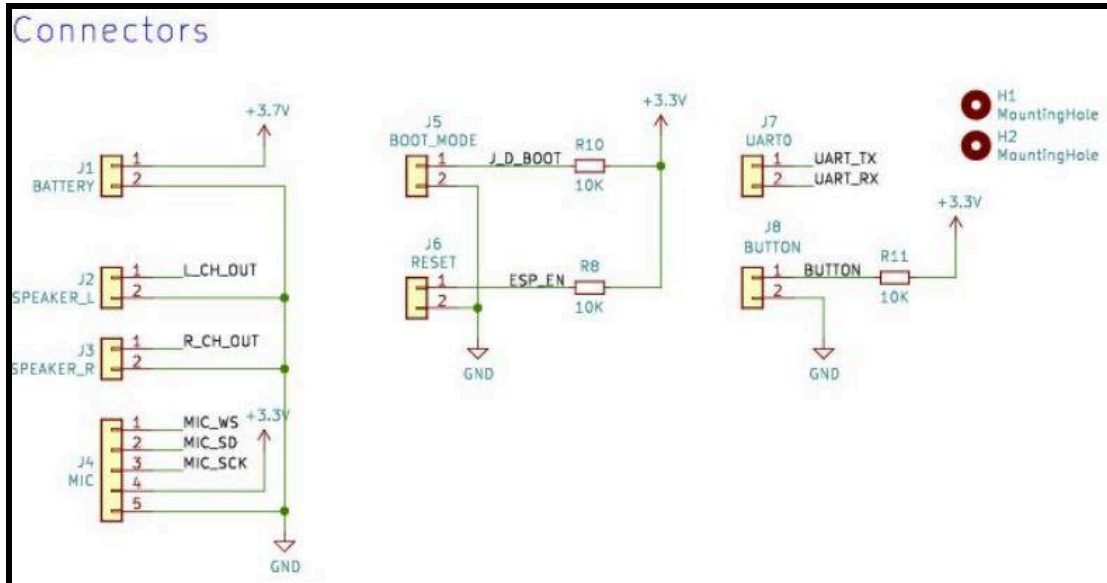


Figure 12: Connectors, Microphone, and Push Button Schematic

## 2.4.2 Functionality and Contribution

Our peripheral subsystem consists of a camera, push button, and speakers, whose schematics are shown above. All components in the peripheral subsystem receive a 3.3-volt signal from the power subsystem to act as their operating voltages. For the camera, we chose to use the Arducam OV2640 CCM, which will communicate with the microcontroller via the I2C protocol. The camera has built-in jpeg encoding to aid with our application subsystem and also has a 1600x1200 color resolution. To ensure the camera can accurately communicate with the microcontroller, we will also implement a camera driver with an external clock set at a 24 MHz frequency. This clock frequency is the typical value for our chosen camera and should ensure proper functionality. A more detailed description of the microcontroller is contained in section 2.5.

The push button in our peripheral subsystem works as a way of commanding the camera to take a photo. The push button will be connected between an active high signal and ground; when the button is pressed, it will send an active low signal to the microcontroller, and when it is not pressed, the microcontroller will receive a low signal. The signal from the button will be integrated within the implemented camera driver to perform the desired camera behavior.

The microphone we chose to implement is the ICS 43434. The microphone outputs digital signals, so it communicates easily with the microcontroller via the I2S interface. The microphone will be connected to the general purpose input/output (GPIO) pins of the microcontroller, which will also have an implemented microphone driver to appropriately handle audio prompts from the user.

The speakers we chose to use for this project are ADS01008MR-LW100-R. These speakers run on the I2S communication protocol and will be connected to the microcontroller's GPIO pins. Another driver for the speakers will be developed and implemented in the microcontroller to ensure adequate handling of audio signals. The microcontroller emits digital audio signals, and the speakers emit audible analog signals, hence the inclusion of a PCM5100APW digital-to-analog converter (DAC) in the speaker DAC schematic (Figure 11). The DAC acts as a bridge between the microcontroller and the speakers, ensuring the speakers will receive the appropriate analog signals. The DAC is also paired with a network of coupling capacitors to mitigate the effects of any potential DC offset and ensure only the desired AC signals are passed through to the speakers.

The purpose of the peripheral subsystem is to communicate with the user and the outside world. This will be done via the pushbutton that the user physically presses, the microphone the user speaks into, the camera that takes photos of the outside world, and the speakers that emit audio to the user. These components will be soldered onto the main PCB and encased within the physical subsystem for protection. The pushbutton or microphone will enact an instance of the camera to take a photo, which will then be fed through the application subsystem. The speakers will then receive and project the appropriate audio via the DAC after the application subsystem is finished with its task.

The peripheral subsystem contributes heavily to the general operation of our project. The camera enables the overall glasses system to have a photo to process, and the speakers and push button enable the user to prompt the system and receive the desired information efficiently.

### 2.4.3 Requirements and Verification

Requirements	Verification
Only the microphone and push button should initiate the glasses system: no other process should start it, and it should not start randomly	<ul style="list-style-type: none"> <li>● Attach an oscilloscope probe at the pin of the microcontroller that is responsible for activating the camera</li> <li>● Test active high inputs to the other pins of the microcontroller coming from the other subsystems</li> <li>● If anything other than inputs from the microphone/push button cause the oscilloscope reading to change, then the test has failed</li> </ul>
Only select, specified prompts to the	<ul style="list-style-type: none"> <li>● Attach an oscilloscope probe at the</li> </ul>

<p>microphone will begin the glasses system. No background noise should start the system</p>	<p>pin of the microcontroller that is responsible for activating the camera</p> <ul style="list-style-type: none"> <li>● Prompt the microphone with similar phrases to those that are supposed to begin the system (similar in syllable count, wording, etc.)</li> <li>● If anything other than correct prompts to the microphone cause the pin to obtain an active high signal, the test has failed</li> </ul>
<p>The speakers must output audio that can be comfortably heard by the user. Research would indicate that this range is around 50 dB</p>	<ul style="list-style-type: none"> <li>● The glasses speakers will be positioned an inch away from a phone</li> <li>● An audio prompt is to be played from the speakers</li> <li>● If the phone's sound level meter detects any sound level outside of the 45-55 dB range, the test has failed</li> </ul>

#### 2.4.4 Design Decisions

We decided to use the Arducam OV2640 CCM due to its color imaging ability and high resolution (1600x1200). With a higher resolution, the ability of our application subsystem to achieve accurate results should be enhanced. When researching cameras that would be compatible with our chosen microcontroller, we also discovered a GitHub with an example driver for the Arducam OV2640 CCM, which we could use as a reference when developing our driver. The decision to use the ICS 43434 microphone and ADS01008MR-LW100-R speakers mainly stems from their use of the I2S protocol. By choosing components that run on the same communication protocol, we can simplify our design. The form factor is also a major constraint for the glasses as we do not want an overly bulky design. To aid with this constraint, we opted to choose physically smaller components. Lastly, we chose components that all have a 3.3-volt operating voltage to simplify our design of the power subsystem.



## 2.5 Power Subsystem

### 2.5.1 Design Overview

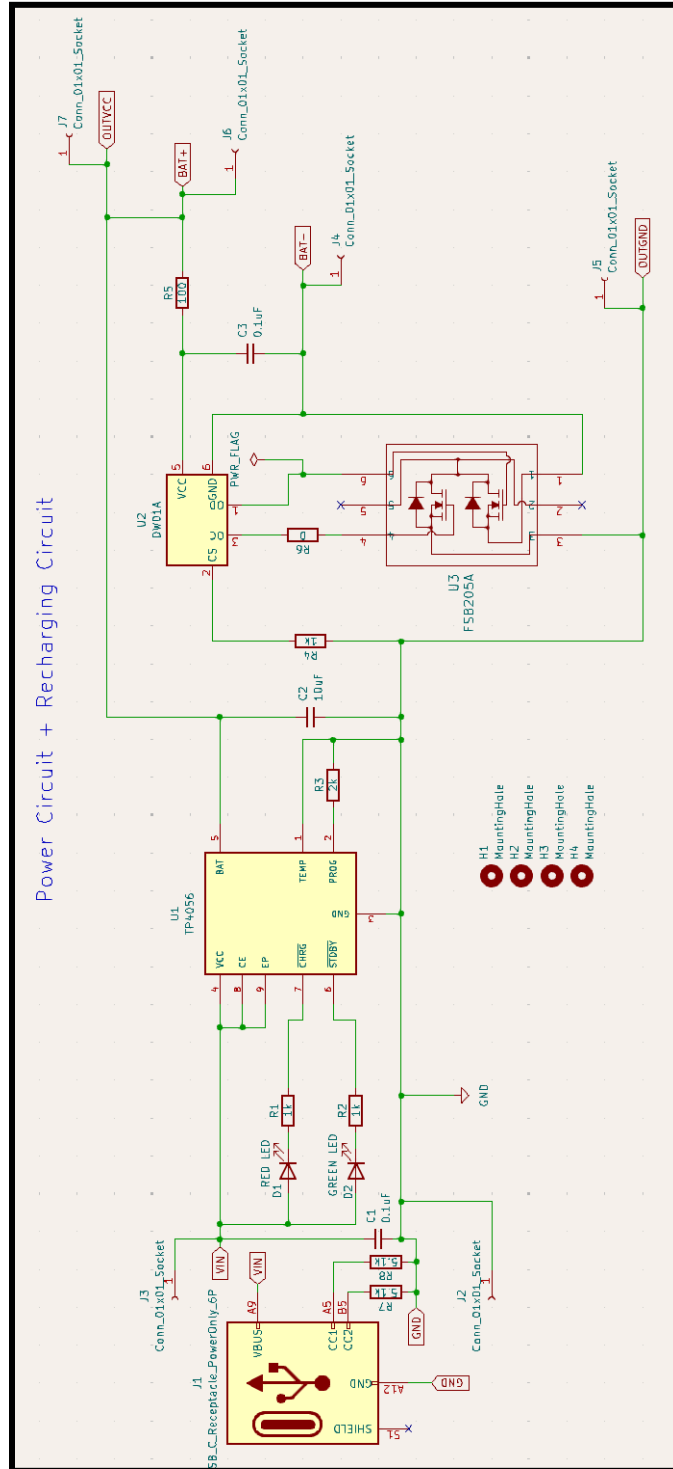


Figure 13: Power Circuit with Recharging Capabilities Schematic, Outputs 3.7V

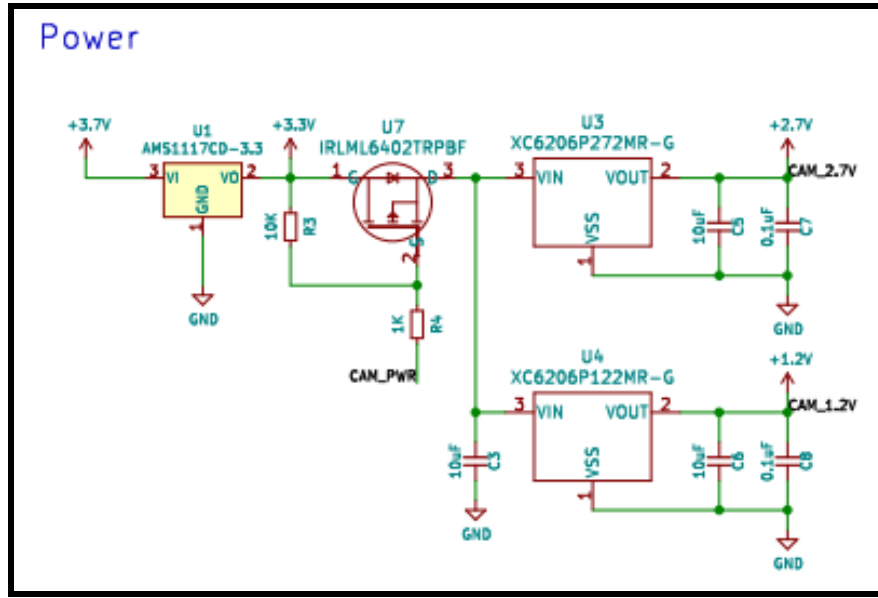


Figure 14: Voltage Regulation Circuit, 3.7V converted to 3.3V, 2.7V, and 1.2V

## 2.5.2 Functionality and Contribution

The power subsystem is responsible for providing 3.3V of regulated voltage to the microcontroller (ESP32-S3-WROOM-1-N16R8), the microphone (ICS 43434), the DAC (PCM5100APW), the camera (Arducam OV2640 CCM), and the push button. Additional linear voltage regulators are also used to output 2.7V and 1.2V to power analog and logic core components in our camera, respectively. It is also responsible for recharging our Lithium-Ion Polymer battery (PRT-18286), along with providing overcharge, over-discharge, and short-circuit protection for our power circuit. The subsystem consists of eight main components that serve individual purposes:

1. The USB\_C\_Receptacle\_PowerOnly\_6P (USB4135-GF-A) connector is used as the power input for charging the battery. It receives 5V from a USB-C power source (Macbook charger), which is then fed into the TP4056 IC. The CC1 and CC2 pins are the plug orientation detection and power delivery negotiation pins. By connecting them to 5.1k Ohm pull-down resistors, the USB-C is identified as a power sink.
2. The TP4056 is a linear lithium-ion battery charging IC that regulates charging current and voltage to safely charge a single-cell 3.7V LiPo battery, like the one we have mentioned previously. Specifically, the TP4056 PROG pin is responsible for the CC phase (constant current charging), and the BAT pin is responsible for switching from the CC to the CV phase (constant voltage charging) once the battery's voltage is at 4.2V. Charging fully stops when the current drops below 10% of the set current, and red and

green LEDs are used to indicate charging and full-charge states of the battery, respectively.

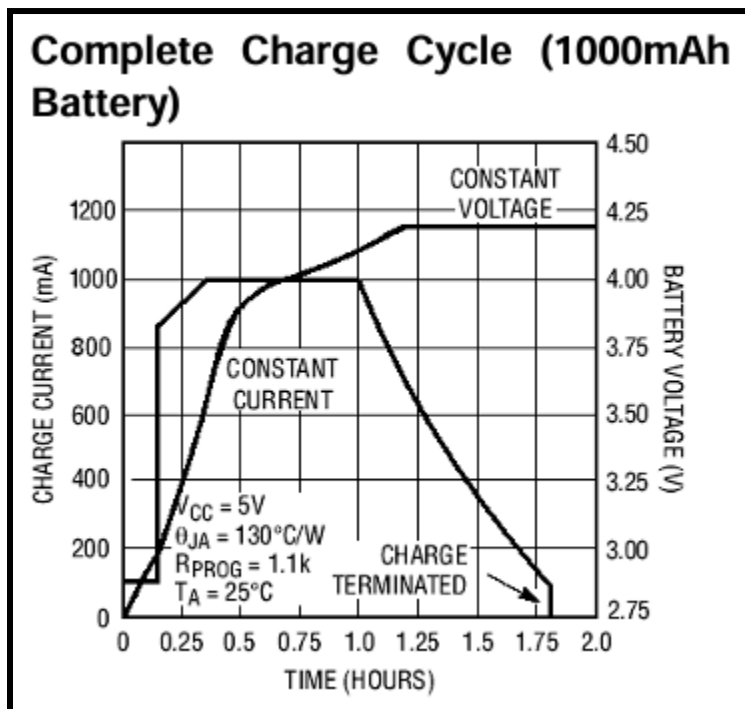


Figure 15: CC/CV charging cycles for a 1000mAh battery from TP4056 datasheet

- The DW01A is a battery protection IC that protects the battery from overcharge (exceeding 4.2V), over-discharge (cuts off the battery if voltage drops below 2.4V), and short-circuit (shuts down the circuit if the current exceeds a safe limit). It does not directly switch power and instead works hand-in-hand with the FS8205A dual-MOSFET to turn charging/discharging on and off. The VCC and GND pins are connected to the positive and negative terminals of the battery as a reference, respectively. For overcharge, if  $V_{CC} > 4.2V$ , the OC pin goes low, turning off charge MOSFET. For over-discharge, if  $V_{CC} < 2.4V$ , OD pin goes low, turning off discharge MOSFET. For short-circuit, if the CS pin senses too high of a voltage difference (indicating excessive current flow) across an external resistor (internal to FS8205A), the DW01A turns off both MOSFETs, effectively cutting all power.
- The FS8205A is a dual-MOSFET that works with the DW01A. It is essentially the switch to disconnect the battery, controlled by the specifications of the DW01A as aforementioned.

5. The AMS1117CD-3.3 is a low-dropout voltage regulator (LDO) that converts its input voltage (supplied from the battery) into a regulated voltage of 3.3V, which will be used for many of our peripherals and also our microcontroller.
6. The IRLML6402TRPBF is a P-channel MOSFET that, in combination with a 1k $\Omega$  pull-up resistor, will allow for the camera to be powered on and off using the ESP32's GPIO. By default, the camera's operation is on due to the pull-up resistor. Once brought low using the GPIO pin, the camera will be powered off.
7. The XC6206P272MR-G is a low-dropout voltage regulator (LDO) that converts its input voltage (3.3V) into a regulated voltage of 2.7V, which is used specifically for powering analog components in our camera.
8. The XC6206P122MR-G is a low-dropout voltage regulator (LDO) that converts its input voltage (3.3V) into a regulated voltage of 1.2V, which is used specifically for powering logic core components in our camera.

When charging our battery, it is important to consider the appropriate resistor for the PROG pin of the TP4056, as this will determine the charging current during the CC phase. The specifications of our battery must line up with the specifications of the TP4056 to prevent any form of power malfunction. The formula given by the TP4056 datasheet is:

$$I_{BATTERY} = \frac{V_{PROG}}{R_{PROG}} * 1200, V_{PROG} = 1V$$

The standard charge current for the PRT-18286 is 0.5C, at a capacity of 1250mAh (on datasheet). This means that the recommended charging current is:

$$I_{CHARGE} = 0.5C * 1250mAh = 625mA$$

Factoring this into our initial equation:

$$I_{BATTERY} = \frac{V_{PROG}}{R_{PROG}} * 1200$$

$$625 = \frac{1}{R_{PROG}} * 1200$$

$$R_{PROG} = 1.92k\Omega$$

Rounding this value to  $2k\Omega$ , we now have the appropriate resistor for the PROG pin of the TP4056, which will result in 600mA of current being used to charge the battery.

### 2.5.3 Requirements and Verification

Requirements	Verification
<p>Must be able to output 3.3V, 2.7V, and 1.2V to our peripherals and microcontroller, independent of one another.</p>	<ul style="list-style-type: none"> <li>● Power circuit and measure the output voltage at each of the LDOs using a multimeter at no load. We should expect to see values of 3.3V, 2.7V, and <math>1.2V \pm 5\%</math>.</li> <li>● Power only one regulator at a time to ensure the other outputs are unaffected. Go through all 8 possible combinations of on and off LDOs to make sure voltage readings are the same as above when on.</li> <li>● Measure voltages when device loads are attached (peripherals and microcontroller). Ensure that we still see the same voltage values as what we saw from the multimeter.</li> </ul>
<p>The battery must be able to be recharged via 5V USB input within 4 hours (based on the PRT-18286 datasheet), and charging should automatically stop when the battery is at maximum capacity.</p>	<ul style="list-style-type: none"> <li>● Start with the battery discharged to 3.0V, the cut-off voltage for our PRT-18286. Monitor the current delivered to the battery with a multimeter, making sure that the current is <math>600mA \pm 5\%</math>.</li> <li>● Verify that during the CC phase, the voltage of the battery is increasing and stops at 4.2V.</li> <li>● Once the battery is at 4.2V, verify using a multimeter that the current being fed into the battery is now decreasing during the CV phase. Charging should fully stop once the current drops below 10% of the set charging current (60mA).</li> <li>● The red LED should be on during the</li> </ul>

	CC/CV phase and on for about 4 hours, while the Green LED should turn on, and the Red LED should turn off once the multimeter reads 0mA.
--	--

#### 2.5.4 Design Decisions

The TP4056, DW01A, and FS8205A were all chosen out of necessity for the charging circuit, along with the cost effectiveness and small size needed for the limited space we have on our glasses. The USB-C receptacle was chosen in our schematic as we already have pre-existing Macbook Pro chargers that can be used to cut costs from having to order a different cable or adapter. Lastly, the LDOs and PMOS were chosen due to their cost-effectiveness and small size compared to a buck converter, which will be talked about in our tolerance analysis.

## 2.6 Communication Subsystem

### 2.6.1 Design Overview

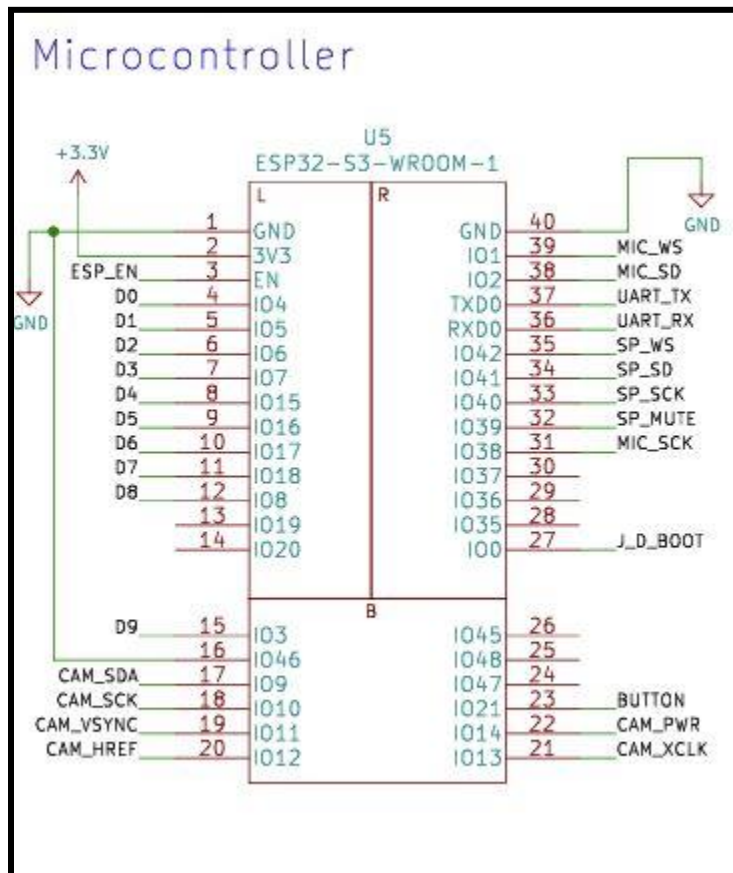


Figure 15: Microcontroller Schematic

### 2.6.2 Functionality and Contribution

Our communication subsystem acts as a bridge between the peripheral and application subsystems. Our microcontroller to handle this task is the ESP32-S3-WROOM-1-N16R8. The GPIO pins of the microcontroller will be wired to the components of the peripheral subsystem (push button, microphone, speaker DAC, and camera). It will communicate with the speaker DAC and microphone via the I2S protocol and with the camera via the I2C protocol. This communication will be enabled by drivers for each respective device. Additionally, the microcontroller will provide a 24 MHz clock signal to the Arducam OV2640 CCM camera to ensure the camera can accurately capture images and transfer them. The microcontroller will be powered by a 3.3V signal from the power subsystem and will communicate with the application subsystem via its built-in Bluetooth antenna.

The purpose of this subsystem is to facilitate communication between the peripheral subsystem and the application subsystem, which contributes heavily to the overall functionality of our design. With this subsystem properly constructed, the camera will only take photos at the desired times (when prompted by the microphone or pushbutton), the application subsystem will receive the captured images via Bluetooth, and the speaker DAC will receive the audio prompt sent from the application subsystem.

This subsystem will also help to achieve our high-level requirement of a fast computation time. Using a microcontroller that can run its core at up to 240 MHz can help speed up the process of image and audio transfer between the peripheral and application subsystems.

### 2.6.3 Requirements and Verification

Requirements	Verification
<p>Communication with all devices in the peripheral subsystem (push button, camera, microphone, speaker DAC)</p>	<ul style="list-style-type: none"> <li>● Camera <ul style="list-style-type: none"> <li>○ After programming, open the serial monitor in Arduino</li> <li>○ Save the information sent over the serial bus to the PC</li> <li>○ If the opened jpeg file is not an image, the test has failed</li> </ul> </li> <li>● Push button <ul style="list-style-type: none"> <li>○ Attach a digital multimeter probe to the pin of the microcontroller connected to the push button</li> <li>○ If the button is pressed and the signal at the pin does not switch to active low, then the test has failed</li> </ul> </li> <li>● Microphone <ul style="list-style-type: none"> <li>○ After programming the microcontroller, add statements within the code to print the value of the audio signal received</li> <li>○ If there is an error printing the value or the value is zero when sound is being made, the test has failed</li> </ul> </li> </ul>



	<ul style="list-style-type: none"> <li>● Speaker DAC <ul style="list-style-type: none"> <li>○ Attach an oscilloscope to the DIN pin of the speaker DAC</li> <li>○ Monitor the oscilloscope reading when data should be sent to it</li> <li>○ If the oscilloscope reading does not change, then the test has failed</li> </ul> </li> </ul>
Bluetooth Connection established with the application subsystem	<ul style="list-style-type: none"> <li>● After programming the microcontroller to use Bluetooth low energy add print statements within a polling loop to print to the serial monitor when it establishes Bluetooth connection with the application</li> <li>● If the serial terminal never prints a connection message, the test has failed</li> </ul>

#### 2.6.4 Design Decisions

We chose to use this particular version of the ESP32 microcontroller (ESP32-S3-WROOM-1-N1698) primarily due to its built-in Bluetooth antenna. Since we wanted to communicate with a mobile app for image processing, the use of this microcontroller eliminated the need for an external antenna. The ESP32-S3 series supports camera interfaces, which is an essential feature in our project. Additionally, the ESP32 is supported by Arduino IDE, which is a platform some of our group is familiar with which should aid in our development of drivers for our devices from the peripheral subsystem.

## 2.7 Application Subsystem

### 2.7.1 Design Overview

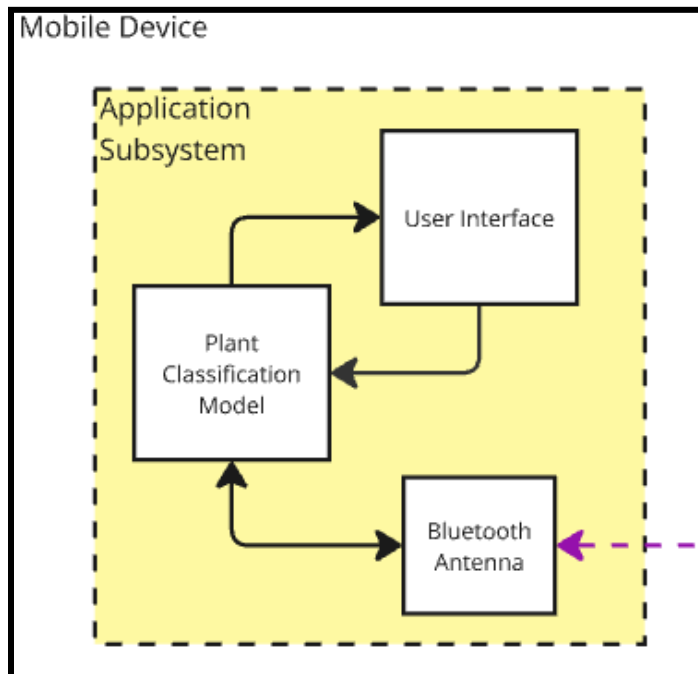


Figure 16: Application Flow Diagram

### 2.7.2 Functionality and Contribution

The application subsystem will interface with the communication subsystem to receive and send data for app functionality. The UI/UX of the application will be on an iOS device and developed using Xcode tools and Swift programming. The application will utilize the mobile device's onboard Bluetooth to send and receive data.

Overall, the application itself will receive the image from the communication subsystem, process the image using a plant classification algorithm (the image processing model that we will use will be discussed in our tolerance analysis), generate an audio stream, and then relay this audio stream through a Digital-to-Analog Converter (DAC). This analog signal will finally be played by the speaker on the glasses, giving the user an audible description of the photo initially taken.

### 2.7.3 Requirements and Verification

Requirements	Verification
--------------	--------------

<p>The application must be able to receive a photo taken by the camera.</p>	<ul style="list-style-type: none"> <li>• The application should show the same picture that was taken by the camera and should be vivid and visible. 10 pictures will be taken by the camera in intervals of 30 seconds to verify that a photo is ready to use for further image processing on the application.</li> </ul>
<p>The application must have a working image processing algorithm.</p>	<ul style="list-style-type: none"> <li>• This verification process is very similar to our high-level requirement verification process.</li> <li>• The algorithm should be able to correctly identify plants 85% of the time. This will be based on our application subsystem, which will use a model to determine the plant's classification. We will test the algorithm on 20 plant images and expect to get 17 out of the 20 correct.</li> </ul>
<p>The application must be able to send accurate data to the Digital-to-Analog converter.</p>	<ul style="list-style-type: none"> <li>• Run the algorithm through our selected classification model 10 times in intervals of 1 minute. Verify that each time this test is run, audible and crisp audio output is being outputted from the speaker, indicating that the application and DAC interface are working as intended.</li> </ul>

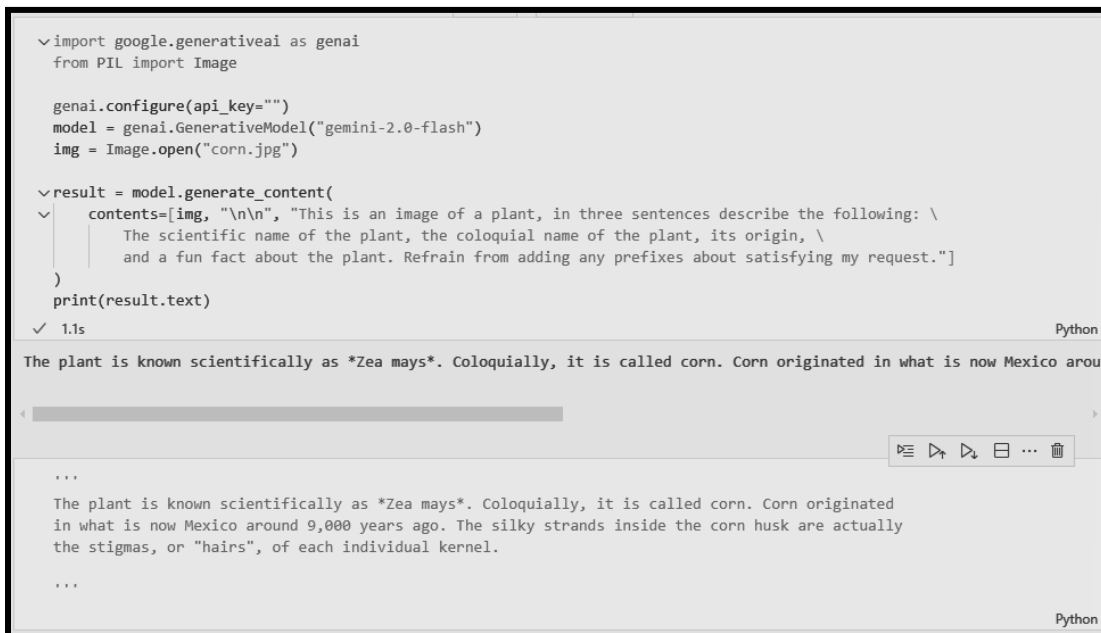
#### 2.7.4 Design Decisions

We decided that an iOS application would be most feasible for our project to take advantage of the ESP32's Bluetooth module for data transfer and, potentially, the CoreML framework for local models. Additionally, most of the members in our group have a good understanding of XCode tools and Swift, which will make the development of the code much more efficient and potentially leave room for more model complexity. Lastly, the classification model that we will be choosing is elaborated upon in our tolerance analysis.

## 2.8 Tolerance Analysis

### 2.8.1 Application Subsystem - Classification Model Selection

We have identified the subsystem with the greatest difficulty and risk of completion to be the application subsystem. There are several avenues from which we can approach the problem of plant classification. The two directions that stand out are running a model to classify the plant locally on the mobile device or outsourcing the classification to a GPT or similar model in the cloud.



```

import google.generativeai as genai
from PIL import Image

genai.configure(api_key="")
model = genai.GenerativeModel("gemini-2.0-flash")
img = Image.open("corn.jpg")

result = model.generate_content(
    contents=[img, "\n\n", "This is an image of a plant, in three sentences describe the following: \
    The scientific name of the plant, the colloquial name of the plant, its origin, \
    and a fun fact about the plant. Refrain from adding any prefixes about satisfying my request."]
)
print(result.text)

```

✓ 1.1s Python

The plant is known scientifically as *Zea mays*. Colloquially, it is called corn. Corn originated in what is now Mexico around

...

The plant is known scientifically as *Zea mays*. Colloquially, it is called corn. Corn originated in what is now Mexico around 9,000 years ago. The silky strands inside the corn husk are actually the stigmas, or "hairs", of each individual kernel.

...

Python

Figure 17: Example Configuration of LLM Response

The first approach would be sending all the information to a GPT that is focused on plant identification. This would be a relatively simple backend implementation since all of the computing will be done in the cloud. However, the main drawback is that processing images in a large model such as GPT can be lengthy and, therefore, delay the response to the user. Additionally, the accuracy of GPT in this context is not well tested and could be a problem for the accuracy high-level goal. Another consideration is the cost of development with GPT. Since it does not have any free version, it could require funds to develop with OpenAI's model. As a result, we could use a free API, such as the one with Gemini.

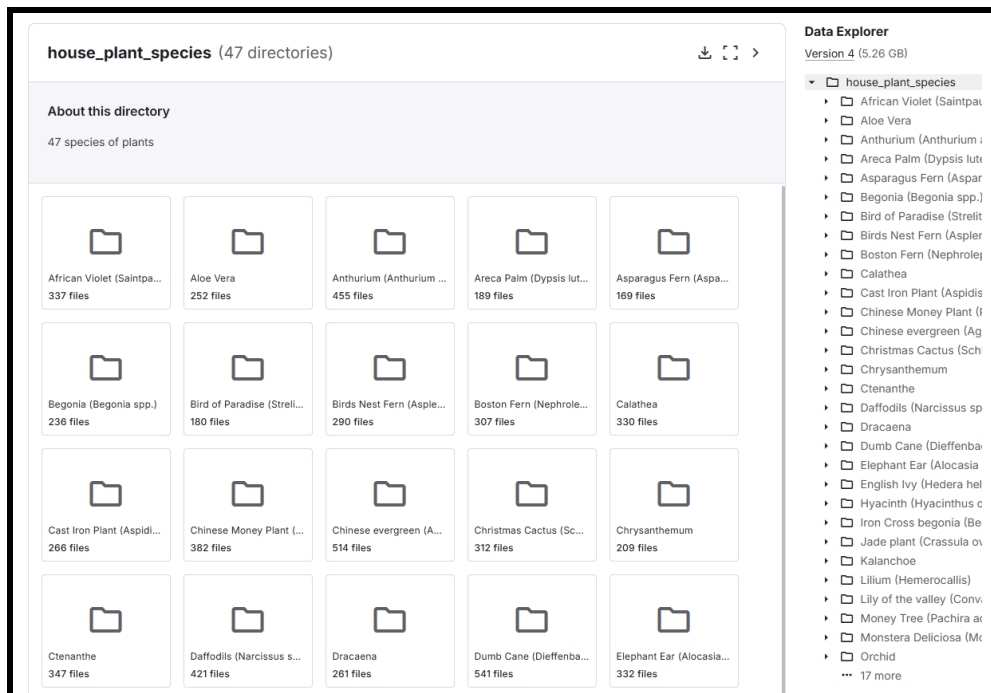


Figure 18: Plant and Label Dataset

The alternative would be creating or finding a model to run locally on the mobile device. This can be done by sourcing a dataset online and utilizing CoreML. Then, more information about the plant could be generated using GPT or pulled from a pre-compiled database. This approach has the upside of being faster since there is less or no information that needs to be processed in the cloud by GPT. Currently, there are not many reliable plant classification models on the market. But there are viable datasets for training our own model. We can use this compiled dataset to create a model and use the CoreML Tools library to convert it to a usable model with Xcode.

To accommodate this potential shortcoming, we might expect a delay of 12 seconds between procuring the plant image and the audio dialogue being played to the user in the case where the local model is too cumbersome to run on the mobile device.

## 2.8.2 Power Subsystem - LDO vs. Buck Converter

As aforementioned, our battery will output a voltage that needs to be reduced to lower voltages for our peripherals' operating voltages. The microcontroller will interface in a similar manner, as the ESP32 has an operating voltage of 3.3V.

However, if the ESP32 is drawing significant current (300mA+ through the Bluetooth Low Energy Module), an LDO reducing voltage from a high 3.7V to 3.3V will waste energy as heat, reducing efficiency. This is different from our peripherals, as they will most likely draw

low currents, in which case the LDO will be efficient. If, in the future, we deem that this power loss is too great, we may change from a generic LDO to a buck converter in order to send a lower voltage to the microcontroller.

If the battery is at a nominal voltage of 3.7V, an LDO drops the voltage by 0.4V for the operating voltage of the ESP32 at 3.3V. Using Ohm's Law and the Power Dissipation Formula and assuming the Bluetooth module will draw a current of 300mA:

$$P_{LOST} = IV$$

$$P_{LOST} = 300mA * 0.4V$$

$$P_{LOST} = 120mW$$

A buck converter steps down voltage efficiently by switching at high frequencies and utilizing an inductor to store and release energy. At an efficiency of 90% for typical buck converters and an assumed 300mA drawn by the ESP32, the power lost would be:

$$P_{OUTPUT} = IV$$

$$P_{OUTPUT} = 300mA * 3.3V$$

$$P_{OUTPUT} = 990mW$$

$$P_{INPUT} = \frac{P_{OUTPUT}}{EFFICIENCY}$$

$$P_{INPUT} = \frac{990mW}{0.9} = 1100mW$$

$$P_{LOST} = P_{INPUT} - P_{OUTPUT}$$

$$P_{LOST} = 1100mW - 990mW = 110mW$$

As we can see, at a high enough current, a buck converter saves more power than a linear voltage regulator (Since our lost power values are very close in this example, anything greater than 300mA would have a greater impact on this analysis). Due to the limited size of our form factor, it is important to prioritize power consumption and physical space. However, LDOs are much cheaper and take up less space than typical buck converters, so the trade-off is only worth it if the microcontroller draws in current greater than 300mA. From online research,

Bluetooth Low Energy typically draws in relatively low current, only upwards of 150mA. Thus, an LDO would be preferable. However, we will test both of these options to see whether or not an LDO or a buck converter for the microcontroller would be better, leveraging the aforementioned tradeoffs to finalize our design decision.

## 3 Cost and Schedule

### 3.1 Cost

#### 3.1.1 Labor

To determine the labor cost of producing our project, we will use the average starting salaries of University of Illinois at Urbana Champaign electrical and computer engineers. The average starting salary for an electrical engineer is \$88,321 [4] and the average starting salary for a computer engineer is \$118,752 [5]. The average number of work days in a year in the United States is 260 with an average work day of 8 hours. This leaves us with approximate hourly wages for electrical and computer engineers of \$42.46 and \$57.09 respectively. Our group is composed of 2 computer engineers and 1 electrical engineer giving us an estimated average group hourly rate of \$52.21. We anticipate that the project will take around 150 hours to complete (50 hours per person), giving us an estimated labor cost of \$19,578.

#### 3.1.2 Parts

Description	Manufacturer	Quantity	Cost
Arducam OV2640 CCM Camera	Arducam	1	\$8.99
ICS-43434 Microphone	Adafruit Industries LLC	1	\$4.95
ADS01008MR-LW1 00-R Speakers	PUI Audio Inc.	2	\$4.32
PCM5100APW DAC	Texas Instruments	1	\$2.50
ESP32-S3-WROOM- 1-N16R8 Microcontroller	Espressif Systems	1	\$6.56
ESP32-S3-WROOM- 1-N16R8 Dev Kit	Espressif Systems	1	\$9.99



B3U-1100P Pushbutton	Omron Electronics Inc-EMC Div	1	\$0.85
PRT-18286 Rechargeable Battery	SparkFun Electronics	1	\$10.95
TP4056 Charging Module	HiLetgo	1	\$5.99
USB-C Charger	Amazon	1	\$8.99
DW01A Overcharge Protection IC	EVVO	1	\$0.23
FS8205A Dual-MOSFET	EVVO	1	\$0.34
USB4135-GF-A USB-C Port	GCT	1	\$0.67
IRLML6402TRPBF PMOS	Infineon Technologies	1	\$0.38
B2B-PH-K-S Connector	JST Sales America Inc.	2	\$0.24
PHR-2 Connector	JST Sales America Inc.	2	\$0.20
FPC FFC PCB Converter Board	MECCANIXITY	1	\$6.89

Parts Total = \$77.36

### 3.1.3 Total Cost

Summing the values obtained for labor and parts, we estimate a total cost of \$19655.36.

## 3.2 Schedule

Week of 3/3:

- Finish Schematic/PCB Layout, Order Main PC for 1st Order - **Nikita**
- Physical Design CAD Model - **Nikita**
- Design Document - **Sunny, Evan, Nikita**
- Teamwork Evaluation I - **Sunny, Evan, Nikita**
- Lab Notebook - **Sunny, Evan, Nikita**

Week of 3/10:

- Finish Schematic/PCB Layout, Order Charging PCB for 2nd Order - **Sunny**
  - Order Power/Charging Parts
- Assemble Main Schematic on Breadboard - **Sunny, Evan, Nikita**
- Assemble Charging Schematic on Breadboard - **Sunny, Evan, Nikita**
- Implement Camera Driver for OV2640 - **Nikita**
- Establish Bluetooth Communication, Receive Image on Laptop - **Evan**
- Lab Notebook - **Sunny, Evan, Nikita**

Week of 3/17:

- Spring Break - **Sunny, Evan, Nikita**
  - Finish up 3/10 Tasks if applicable

Week of 3/24:

- Barebones Application UI Creation - **Sunny**
- Debug Circuits (if necessary) - **Sunny, Evan, Nikita**
- Assemble/Solder Main and Charging PCBs - **Evan, Nikita**
- Individual Progress Reports - **Sunny, Evan, Nikita**
- Lab Notebook - **Sunny, Evan, Nikita**

Week of 3/31:

- Implement I2S Drivers - **Evan**
  - Speaker DAC
  - Microphone
- Test Classification Methods (Tolerance Analysis) - **Sunny, Nikita**
  - Plant Classification with GPT
  - Local Model/Database with CoreML
- Establish Communication Between App and Microcontroller - **Sunny, Nikita**

- Order revised PCBs for 3rd Order (if necessary) - **Sunny, Evan, Nikita**
- Individual Progress Reports - **Sunny, Evan, Nikita**
- Lab Notebook - **Sunny, Evan, Nikita**

Week of 4/7:

- Order revised PCBs for 4th Order (if necessary) - **Sunny, Evan, Nikita**
- Glasses Communication Framework - **Evan, Nikita**
  - Gives App Access to Peripherals
- Interface Plant Classification App to Peripherals - **Sunny, Nikita**
  - Request Image Upon Button Activation
  - Play Audio Data to Speakers
- Print 3D Frame Model - **Evan**
- Lab Notebook - **Sunny, Evan, Nikita**

Week of 4/14:

- Assemble prototype - **Sunny, Evan, Nikita**
- Practice for mock demo - **Sunny, Evan, Nikita**
- Team Contract Assessment - **Sunny, Evan, Nikita**
- Lab Notebook - **Sunny, Evan, Nikita**

Week of 4/21:

- Mock demo - **Sunny, Evan, Nikita**
- Lab Notebook - **Sunny, Evan, Nikita**

Week of 4/28:

- Final Papers - **Sunny, Evan, Nikita**
- Final Demo - **Sunny, Evan, Nikita**
- Practice for final presentation - **Sunny, Evan, Nikita**
- Lab Notebook - **Sunny, Evan, Nikita**

Week of 5/5:

- Final Presentation - **Sunny, Evan, Nikita**
- Final Papers - **Sunny, Evan, Nikita**
- Lab Checkout - **Sunny, Evan, Nikita**
- Lab Notebook - **Sunny, Evan, Nikita**
- Award Ceremony - **Sunny, Evan, Nikita**

## 4 Ethics and Safety

### 4.1 Ethics

A major ethical concern associated with our project is that of privacy due to the inclusion of a camera in our design. The IEEE Code of Ethics compels us to protect the privacy of others [1], which will impact how our application subsystem handles the images taken with the camera. To comply with this legislature, our application will not store any of the images taken with the camera therefore preserving the privacy of anyone photographed with our design. Another potential location where ethics can be involved in the development of our project is when testing. For any tests where we survey a general audience to receive feedback, we are compelled by the IEEE Code of Ethics to treat everybody surveyed fairly with respect and to not engage in any discrimination [1]. By doing this, we will not only comply with the code of ethics but also likely gain more honest feedback if we survey at random and not only people who provide only positive feedback.

### 4.2 Safety

A safety concern associated with our project is water resistance. Since our design is meant to be worn outside, we intend it to be worn in moderate weather conditions, such as light rain. Without proper water resistant treatment, the electrical components could become prone to damage via moisture, which could result in short circuits [2]. This would not only impact the performance of our design but, in extreme cases, could also put the user at risk of receiving an electric shock. To mitigate the possibility of this occurring, we aim to coat the final design with a water-resistant layer.

Another consideration comes with the use of rechargeable lithium-ion batteries in our power subsystem. The use of these batteries can be dangerous as if they are damaged, they have the potential to catch fire [3]. To try and prevent damage to the batteries, we plan to encase them completely within our 3-D printed glass frame design. Additionally, the batteries can reach high temperatures when charging and discharging (45-60 degrees Celsius). Since the batteries are going to be in close contact with the user during use, we plan to use a heat sink combined with small ventilation holes within the 3-D printed glasses frame to dissipate the heat to minimize the risks of burns or even mild discomfort.

## 5 References

### 5.1 Datasheets

Arducam OV2640 CCM - Camera

[https://www.uctronics.com/download/OV2640\\_DS.pdf](https://www.uctronics.com/download/OV2640_DS.pdf)

ICS-43434 - Microphone

<https://invensense.tdk.com/wp-content/uploads/2016/02/DS-000069-ICS-43434-v1.2.pdf>

ADS01008MR-LW100-R - Speaker

[https://www.mouser.com/datasheet/2/334/ADS01008MR\\_LW100\\_R-1916757.pdf](https://www.mouser.com/datasheet/2/334/ADS01008MR_LW100_R-1916757.pdf)

PCM5100APW - Audio Stereo DAC

[https://www.ti.com/lit/ds/symlink/pcm5100a.pdf?ts=1739433553738&ref\\_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FPCM5100A](https://www.ti.com/lit/ds/symlink/pcm5100a.pdf?ts=1739433553738&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252FPCM5100A)

ESP32-S3-WROOM-1-N16R8 - Microcontroller

[https://www.espressif.com/sites/default/files/documentation/esp32-s3-wroom-1\\_wroom-1\\_u\\_datasheet\\_en.pdf](https://www.espressif.com/sites/default/files/documentation/esp32-s3-wroom-1_wroom-1_u_datasheet_en.pdf)

B3U-1100P - Pushbutton, Ultra-small Tactile Switch (SMT)

[https://omronfs.omron.com/en\\_US/ecb/products/pdf/en-b3u.pdf](https://omronfs.omron.com/en_US/ecb/products/pdf/en-b3u.pdf)

PRT-18286 - Rechargeable Battery

[https://cdn.sparkfun.com/assets/e/3/3/2/7/-KPL623450-1300mAh-3.7V\\_\\_\\_\\_\\_2021-01-21.pdf](https://cdn.sparkfun.com/assets/e/3/3/2/7/-KPL623450-1300mAh-3.7V_____2021-01-21.pdf)

TP4056 - Standalone Li-Ion Battery Charger

<https://dlnmh9ip6v2uc.cloudfront.net/datasheets/Prototyping/TP4056.pdf>

DW01A - Overcharge Protection IC

<http://www.tp4056.com/d/dw01a-fs.pdf>

FS8205A - Dual-MOSFET

[https://www.mpu51.com/mcucity/DATA\\_PDF/FS8205A-DS-12\\_EN.pdf](https://www.mpu51.com/mcucity/DATA_PDF/FS8205A-DS-12_EN.pdf)

USB4135-GF-A - USB-C Port

<https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/6501/USB4135%20Product%20Spec.pdf>

IRLML6402TRPBF - PMOS

<https://www.infineon.com/dgdl/irlml6402pbf.pdf?fileId=5546d462533600a401535668d5c2263c>

B2B-PH-K-S - Connector

<https://www.jst-mfg.com/product/pdf/eng/ePH.pdf>

PHR-2 - Connector

<https://www.jst-mfg.com/product/pdf/eng/ePH.pdf>

FPC FFC PCB - Converter Board

<https://media.digikey.com/pdf/Data%20Sheets/Hirose%20PDFs/FPC,FFC%20Connectors.pdf>

## 5.2 Ethics/Safety

[1] IEEE. “IEEE Code of Ethics.” (2020), [Online]. Available:

<https://www.ieee.org/about/corporate/governance/p7-8.html> (visited on 03/06/2025).

[2] Conro. “How to Waterproof Electronics?” (2023), [Online]. Available:

<https://www.conro.com/Blog/How-to-waterproof-electronics/> (visited on 03/06/2025).

[3] L. Kong, C. Li, J. Jiang, and M. Pecht, “Li-ion battery fire hazards and safety strategies,” *Energies*, vol. 11, p. 2191, 08 2018. [Online]. Available:

<https://www.mdpi.com/1996-1073/11/9/2191>

[4] G. E. O. of M. and Communications, “Electrical Engineering,” *grainger.illinois.edu*.

<https://grainger.illinois.edu/academics/undergraduate/majors-and-minors/electrical-engineering>

[5] G. E. O. of M. and Communications, “Computer Engineering,” *grainger.illinois.edu*.

<https://grainger.illinois.edu/academics/undergraduate/majors-and-minors/computer-engineering>