Team A7: deciBright

Authors: Lucy Chen, Katherine Sabak, Freda Su
Affiliation: Electrical and Computer Engineering, Carnegie Mellon University

***Abstract*—For musicians and others who need to track their exposure to noise over time, the deciBright bracelet and accompanying mobile web application is a fast visual solution for measuring and logging decibel levels. The bracelet changes color in response to loud sounds within 1 second.**

***Index Terms*— BLE, Decibel, LED, Microcontroller, Sound Volume, Web Application**

1 INTRODUCTION

According to research recently published in Ear and Hearing, "hearing loss was the most prevalent sensory disorder in the United States" from 1990 to 2019 [4]. Other than obvious uses such as listening to music and friends, our ears are also important to help with balancing, and even mild hearing loss doubles the risk of dementia [20]. Thus, it is important to protect our hearing through minimizing exposure to loud sounds, and by becoming more aware of potential risks in the environment. While people can discern some relative volume, the ears are actually not always accurate at determining the "loudness" vs decibel value, because of sensitivity to certain frequencies [29].

Existing decibel meters and phone apps can give more precise information, but they are still disruptive to pull out and check throughout the day. A wearable device would be a convenient visual reminder for long-term tracking of volume levels that will leave your hands free. Our solution is a LED bracelet that changes color based on the decibels readings of the surroundings. With the accompanying web app, users can view statistics about their exposure over time, modify the sound threshold levels to suit their needs, and customize the colors to make the bracelet a fashionable accessory. From musicians monitoring their practice sessions, to patients collecting data to assess their risk levels, to anyone who wants to be more mindful about their hearing in general, this bracelet enables users to make more informed choices regarding their auditory health.

The intended user of this product is a student or professional in the field of classical music. Musicians often practice or perform music for up to multiple hours a day, and practice rooms are not always set up to have the best acoustics for their needs. Particularly for vocalists, there is a difference between sounds perceived as loud (including quiet sounds with much resonance) and objectively loud sound, which makes it difficult for someone to evaluate the decibel levels of their surroundings. According to conversations with Dr. Dueck, classical musicians generally do not use in-ear monitors or other digital methods of perceiving feedback about their music, and they need to be able to hear all details of their music. Therefore, hearing protection is particularly important.

Competing technologies include existing decibel meters, which are large and awkward to carry around, and decibel meter apps, which are inconvenient to pull out your phone for, and may prevent you from being able to use your phone for other tasks at the same time. In addition, apps may not be calibrated well, so they aren't the most accurate. The goal of this system is to both provide the user with fast visual feedback about sound levels in their environment (bracelet) and to track data about sound levels over extended periods of time (webapp).

2 USE-CASE REQUIREMENTS

2.1 Weight

The weight of our bracelet should not prevent the user from accomplishing tasks, including those that require precise motions, like playing an instrument. Therefore, our product should be no heavier than the average smart device that could instead be accompanying these tasks: a smartphone. The approximate maximum weight for a smartphone is 200g [8], so that is also this device's maximum weight.

2.2 Width

In order for our device to be an adequate solution, it must closely mimic the physical attributes of existing items with similar forms. To that effect, the width of this bracelet should be no larger than the largest width commonly offered for a wristwatch: 46mm [23]. The results of a poll of music studio students (Fig. 5) showed that a majority would prefer a maximum size of 44mm; this option was likely a mistake by the creator of the poll (46mm was the intended choice), but it suggests that smaller widths are preferred.

2.3 Thickness

Similarly, the bracelet should conform to existing sizing choices for commercial non-electronic bracelets. Our resource on wristwatch sizes [23] describes large bracelet thicknesses as around 9mm; since the plastic insulation on the bracelet slightly extends its size, this requirement is relaxed to be interpreted as a maximum of 12mm.

2.4 Operating Temperature

Since the bracelet will be in contact with the user's skin, we want to eliminate the risk of burns from the product's operating temperature. For our bracelet to be comfortable and safe, we require it to operate at a temperature no higher than 105 °F, well under the human pain temperature threshold of 107.6 °F [27].

2.5 Accuracy

The purpose of our bracelet is to record the sound levels of the environment. Therefore, it needs to accurately record the surrounding noise level. We require the bracelet and app to be within 2 decibels of the actual volume of the room, which is the national standard for sound measurement instruments [12], as measured by a decibel meter.

2.6 Timeliness

Since our bracelet will be measuring noise changes in the environment, we want our bracelet to capture and respond to the most up-to-date sounds. Therefore, we require a latency of no more than 1 second between the detected sound to the webapp and the bracelet color display.

2.7 Adjustability

We want our bracelet to fit comfortably on as many wrists as possible. Therefore, we require an adjustable strap of 7-10 inches (160-228.6mm), which is the standard range of adult bracelet sizes [18].

2.8 Durability

The bracelet should be durable enough to survive a 2.5 feet drop, which is the average height of a standard table [15]. This is to ensure that the bracelet can withstand a certain amount of impact and still function properly. It also reflects a likely type of scenario that the bracelet would encounter. Survival is defined as being able to process microphone system output and display colors in a way that matches its environment (displays same accuracy and responsiveness as before).

2.9 Battery Life

The bracelet should have a battery life of at least 4 hours. This was based on musician survey data, where most practice sessions last less than 4 hours.

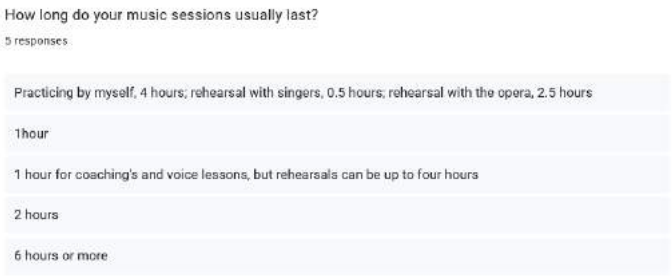

Figure 1: Results of a survey sent to music studio members: most music sessions last 4 hours or fewer.

3 ARCHITECTURE AND/OR PRINCIPLE OF OPERATION

3.1 System Architecture

For our bracelet, the default LED colors will be green, yellow, and red. A green bracelet will mean that the current sound levels are not harmful and will cause no damage to the ear. Yellow would represent a sound level corresponding to 80-100 dB, which may cause damage if exposed for 8 hours or more [16]. Lastly, a red bracelet would represent a sound level of over 100 dB, which can cause hearing damage almost immediately [16]. These colors and thresholds can be customized on the mobile web application.

Important bracelet hardware components include the microcontroller with built-in BLE module, LEDs, microphones, boost converter, and battery. The battery powers the system through a boost converter. A switch in series with the battery-Beetle connection allows the system to be powered on/off. The Beetle microcontroller provides power for the microphones, and connects to the following devices: the LED array via an Arduino digital connection, and microphones via an Arduino analog connection. All components except the battery, buttons, and boost converter are mounted on a custom-designed flexible PCB (Fig. 4).

Since the bracelet will only display colors that represent a range of decibels, it is accompanied by a mobile web application that provides further detail about the user's sound environment. The app is built using React Native. It displays information such as the current decibel level reading and the maximum decibel reading for the session, as well as over time. Lastly, there is a place on the app to customize the intensity of the LEDs, the colors of the bracelet, and the thresholds to be used for each color (Fig. 3).

3.2 Changes from Design Report

Our final system is significantly different from the design report. Since there is no BLE connection between the bracelet and the webapp, features and tests related to this connection have been removed. The webapp also has no option for visualizations or graphs on statistics relating to sound exposure over time. Orange is no longer an offered default color for the LEDs. Three options, representing safe

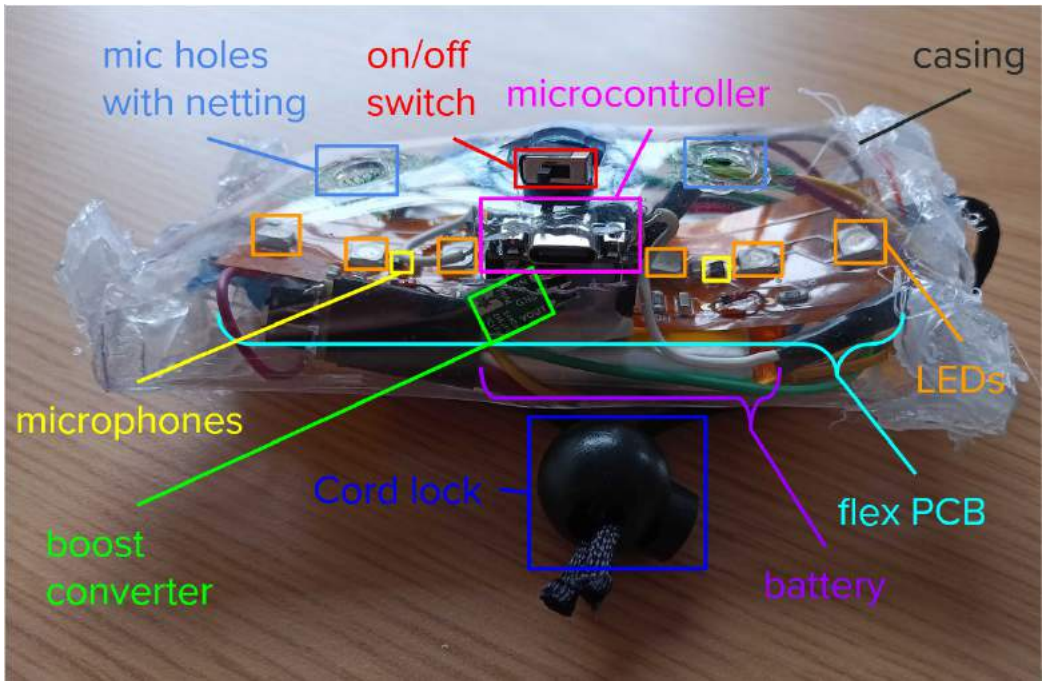

Figure 2: Annotated features of the physical system (bracelet)

levels of sound (below 80 dB), a level of caution (between 80 and 100 dB), and a level of danger (100 dB or higher, are shown respectively as green, yellow, and red.

The LED connection to the Beetle is digital, and the microphone connection is analog; this was reversed unintentionally in the design report.

Since average mode is no longer a feature of our system, the GPIO mode control button (intended to switch between average and instantaneous modes) is not included; the system is always in instantaneous mode, in which the LED colors correspond to the sound heard at that instant in time.

We are also no longer calculating directionality, because of two reasons. First, soldering tiny mics is difficult to do correctly, so we may not be able to have both mics working often. Second, the mics aren't sensitive enough to pick up a difference in sound level that wouldn't be within the margin of error of "noisy readings".

3.3 Principles of Operation

3.3.1 Engineering Principles

Designing and assembling this system required several principles of engineering. The broadest takeaway was the necessity of unit-testing each individual element of the system. For example, to determine why the LEDs on the bracelet did not light up, we checked voltages at many connections between the battery input and LED output, such as at the voltage regulator, and at relevant wires. To check if the mics were working, we compared outputs from the oscilloscope and the Arduino serial plotter. Finally, we ran example code [5] on the Beetle as a "sanity check", such as the "blink" test for the onboard LED, or the "strandtest" that cycles through different LED colors before integrating or modifying existing code. We went back to trying these examples whenever something mysteriously malfunctioned, to help narrow down what part of the system it was.

Other engineering principles included having backups of our product: we created several copies of each version of the PCB, which was helpful when one broke, and to especially to increase our expected value for successful mic soldering since they were so small and difficult to place correctly.

This also allowed us to test variations to narrow down on the best version. For example, the microphone datasheet [3] suggested a range of resistor values could be used, but we didn't know what would work well beforehand. Thus, our first board version used through-hole resistors so we could more easily swap out different resistors/add resistors in parallel/series. Once we settled on a value, then we could convert to SMD resistors for a more efficient soldering process.

Multiple copies also helped with iterating on different ideas to find unconventional solutions. For example, stacking SMD resistors to achieve their combined resistance value because the desired value was unavailable as a single resistor, moving PCB paste with tweezers to correct soldering mask errors, and adding protoboard pieces or hot glue to the reverse side of soldered header pin connections to increase stability on the flex PCB (more details in 5.2.3 and 6.1.2).

3.3.2 Scientific Principles

The principles of science used in our system relate to electronic physics and biology.

For electronics, we use microphones that output a DC base voltage with an additional AC signal corresponding to the sound input (as can be seen on an oscilloscope). The peak detector system uses an RC circuit, which has a time constant that changes depending on the capacitance and resistance values chosen.

On the biology side, our use-case is driven by the fact that human perceptions of "loudness" aren't necessarily the same as decibel levels due to equal-loudness contours [29], where sounds at different frequencies and at different dB levels can sound the same in "loudness" to us. This relates to why sound frequency weightings exists: for example, A-weighting is supposed to match human-perceived loudness levels [2]). Finally, we researched the health side of our ears, such as recommended limits for decibel values [16].

3.3.3 Mathematical Principles

Mathematical principles used in our system generally concern calculating numbers. The most apparent mathematical principle relevant to our project is that the decibel scaling is nonlinear [28].

This makes it reasonable for us to use a best-fit logarithmic function to find the equation determining the dB/Pa value for a known microphone gain resistance. Several gain resistance-dB/Pa pairs are already provided in the microphone datasheet [3]; modeling the effect of the resistance on the dB-Pa value allows us to extrapolate situations not covered in the datasheet, such as the resistor value we ended up choosing.

Other mathematical principles at play in our system include calculating the circumference of the bracelet casing from a known diameter (the PCB board width), because the casing needs to be cut from a "flat" piece of plastic, so this would calculate one of the sides of a rectangle that can wrap around the board.

Finally, averaging peak detector output values to create a baseline startup value helps to solve the problem of the microphones starting off at very different values for seemingly the same environment. If we didn't calibrate, that would mess up the validity of the threshold constants we set.

The application of all these principles is covered in greater detail in the system implementation section.

4 DESIGN REQUIREMENTS

4.1 Operating Temperature

This quantitative requirement does not relate to specific temperatures because we do not have a temperature sensor available, so we can only assess qualitatively from touching it. It should be possible for someone to touch the bracelet while it is in use and not feel uncomfortable from any heat it may be giving off.

4.2 Accuracy

Over a 5-minute testing period, the difference between the webapp decibel reading and decibel meter reading, read in increments of 1 second, will not exceed 2 dB. The decibel meter will be placed close to the bracelet for this test.

4.3 Timeliness

When a new sound volume is introduced to the setting that meets a threshold level required for the bracelet to change color, the bracelet color will respond by changing color in no more than 1 second.

4.4 Adjustability

The bracelet will be adjustable to fit a circumference of 7-10 inches (160-228.6mm). We will use a ruler or measuring tape to check the circumference when bracelet is at its minimum and maximum length after adjusting.

4.5 Durability

The bracelet should be able to operate standard Bluetooth, microphone input, and LED output functions before and after a 2.5ft fall from rest.

4.6 Battery Life

After being turned on, the bracelet should maintain a connection with the webapp and update based on both direct input and its surroundings for at least 4 hours.

5 DESIGN TRADE STUDIES

5.1 Web Application

For our web application, we wanted to create an intuitive, user-friendly interface that is easy to navigate. We have chosen mobile development over web development to allow users the ability to conveniently connect to the bracelet at any time. For our mobile application, we want a framework that can be used on both iOS and Android platforms. The two options we considered were Flutter and React Native, which are both popular open source software frameworks that allow for seamless cross-platform app integration. Ultimately, we chose React Native because its programming language, JavaScript, is the one our group is most familiar with. Flutter uses the programming language Dart, which is a language no one in our group has used before, meaning there would be a learning curve; and with limited time, we decided not to use it.

For Bluetooth connection, there were two React Native libraries to choose from: ble-plx and ble-manager. We switched around a few times between these two libraries. At first, we chose ble-plx because it was the most full-featured Bluetooth library available, with full ability to scan, connect, communicate, and filter for devices. It also

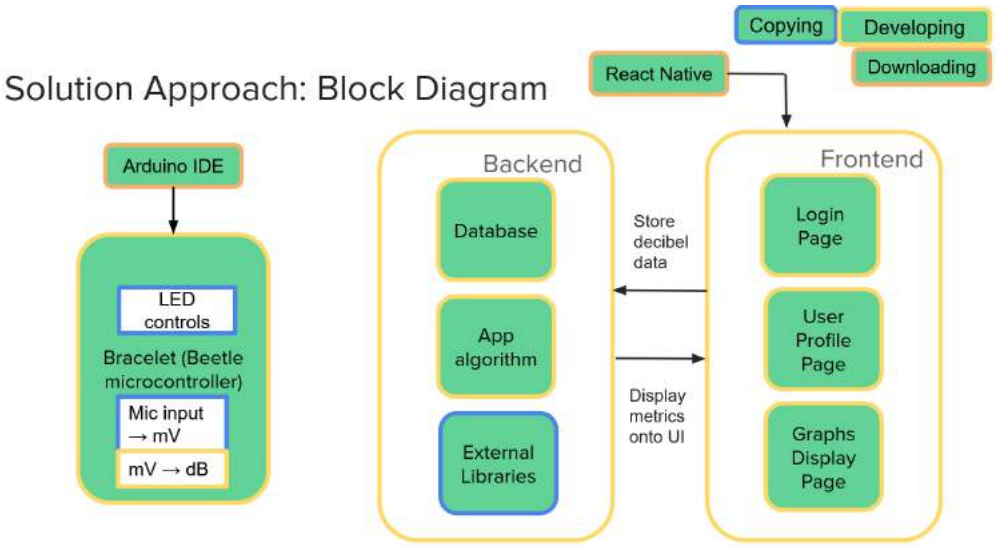

Figure 3: Software system diagram

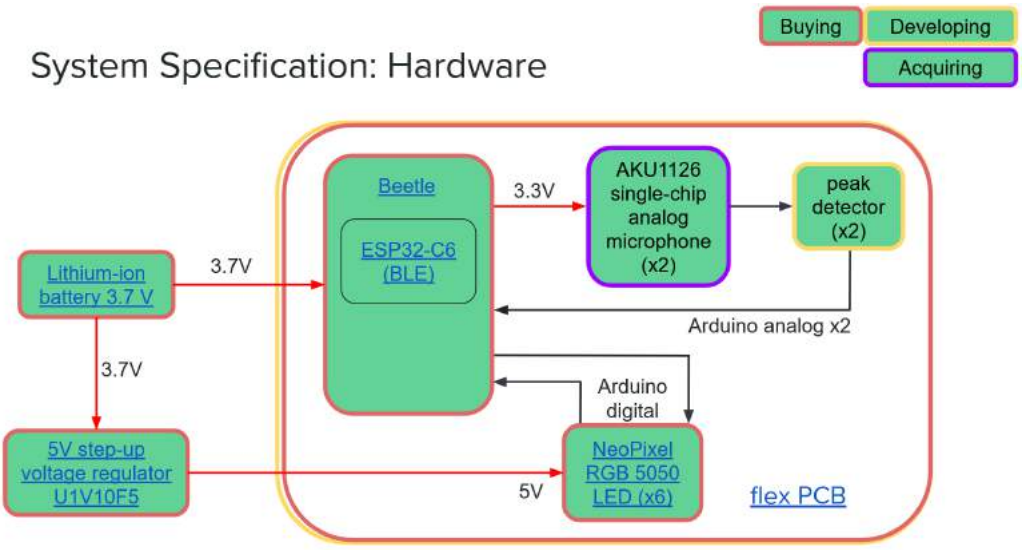

Figure 4: The types of connections between the electronic components are specified.

had good documentation and a huge community of users. However, we found out that ble-plx only supported up to SDK 49, while the rest of our app was on SDK 50. This caused us to switch to ble-manager, which has less functionality than ble-plx but still could provide us with everything we needed. Unfortunately, there were compatibility issues with ble-manager as well, since using this library caused the app to crash.

5.2 Hardware

5.2.1 Battery Life

In choosing to use a battery with a 4-hour life rather than an 8-hour life, we prioritized user comfort over session longevity, based on the majority votes from music students of what they preferred (Fig. 5).

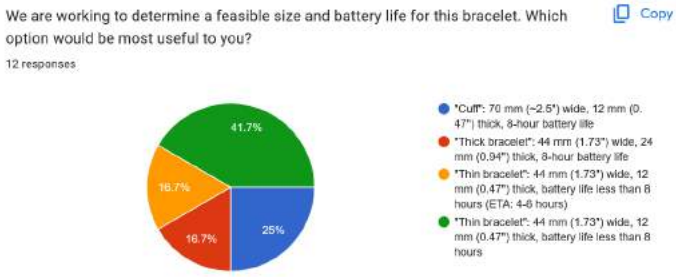

Figure 5: Results of a survey sent to music studio members; over half preferred a bracelet with a battery life of less than 8 hours.

5.2.2 Component Attachment

In primarily using SMD rather than through-hole components, we prioritized a small size and ease of soldering more components at once over the ability to switch out components. This meant we had to be more sure of what resistor and capacitor values we were going to use earlier in the process, which we were able to achieve. We also had a backup plan for changing values last minute if they weren't values available in the PCB lab already, though we didn't end up needing to do that (6.1.2).

5.2.3 PCB Type

In using a flex PCB, we prioritized the circuit board's flexibility, over low cost and technical ease of use. We wanted the bracelet lights to show up around the circumference of the wrist, which was why a flexible board was necessary, rather than a "watch" style light. It also allowed us to arrange wires better since the board could bend around them.

The flex PCB's technical difficulties arose in how thin and light it was; this made solder paste harder to apply lightly, and the oven fans would be able to blow it away.

Our solution was to apply less solder paste than we would for hard PCBs, as well as taping everything down more firmly so less solder paste could get paste the edges they weren't supposed to. We also used tweezers to to clean up smaller problem areas to avoid needing to reapply solder paste to the whole board.

To prevent the flex boards from flying away in the oven, we tied the corners down to the tray with wires through the helpfully provided holes along the edges (Fig. 6).

Finally, post-soldering, we discovered that the header pin where the voltage regulator attachment was supposed to go (and button header pin) was easily ripped out of the board (along with any hope of connecting the traces), so we attached protoboard pieces or hot glue to the area to stiffen and hold the header pin in place. We didn't pull out any pieces after that (Fig. 7).

Thus, our technical issues were solvable with practice and thinking of creative solutions.

Figure 6: Flex PCB secured with wire

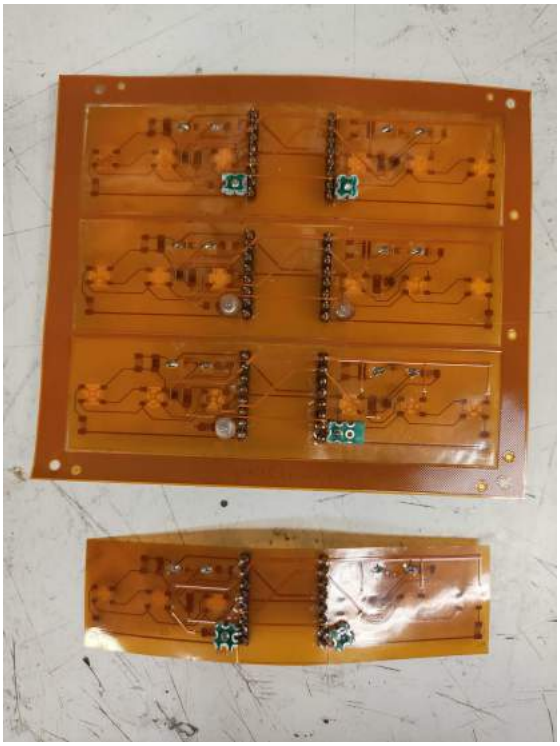

Figure 7: Protoboard pieces and hot glue protecting the header pins

5.2.4 Microphone System: Peak Detector

We had a large debate over whether it was worth adding a peak detector to our system when we didn't have that much time left to order new flex PCBs (which would also need more time for assembling and testing, as well as to practice working on flex PCBs due to the issues we would find in 5.2.3).

The problem was the mics didn't seem to be picking up on sound reliably. However, we didn't know if the peak detector would be reliably better either. Our solution was to quickly prototype it with a breadboard, and compare it to the readings without the peak detector. We found that the peak detector version picked up on sound much more consistently than the raw output from the mics, which convinced us it would be worth the time to redesign the PCBs. As you can see from Fig. 8, the raw mic output picks up on less sound than the peak detector, as well as the range of mV being smaller.

Figure 8: The top 2 wavy lines show the mic's raw output, while the bottom 2 wavy lines show the peak detector output. The straight lines were just there to keep the graph axis from resizing automatically, which Serial Plotter will do if the range of values changes. From a longer video: `https://youtu.be/_pMX454oqqs`.

5.3 Bracelet Assembly

5.3.1 Tube vs Plastic Sheet

We were considering using a plastic tube because we would be doing less cutting and gluing than for a case made out of a flat plastic sheet. The plastic sheet ended up winning due to its customizability for size, as well as being thinner. More of detail of the prototyping tests is in the Casing Subsection (6.3), which helped us determine that the plastic sheet is the clear winner.

5.3.2 Cord Type

We decided we should use some string to tie the bracelet ends, as well as provide adjustability. Ideally, the string would be strong (we don't want the bracelet to snap), resistant to fraying (long-term wear and tear durability), and pliable (to conform well to the shape of the wrist). See Fig 9.

There are many different materials used for bracelets, such as embroidery floss (cotton), polyester cord, and elastic.

	Pros	Cons
Cotton	• Easy to make tight knots ○ Can combine threads for strength • Pliable	• Fraying threads
Polyester	• Smooth, resistant to fraying (melt the ends) • Pliable • Decently strong	• Might not be able to make as-tight knots because of its smoothness/thickness
Elastic	• No need to attach separate ends of the bracelet with a separate mechanism; could just make it 1 circular piece • Pliable • Strong and fray-proof	• Loses elasticity over time; would have to replace • Not as adjustable; depending on the length it could be pinching or still too loose

Figure 9: Cord Results

There wasn't much competition for this one. Since knots aren't really necessary for the bracelet fastening design and strength, and we value long-term usage, polyester won. The thickness of the cord was based on the cord lock dimensions/recommended thicknesses. I chose the lower end of that for most ease of threading it through, since the thicknesses should all be strong enough.

5.3.3 Adjustability Mechanism

The two ends of the bracelet need to be attached in a way that is easily adjustable. Common bracelet attachment/adjustment methods include chain & clasp, pulling on both ends (specific knotting technique), and cord lock mechanism. See Fig 10.

	Pros	Cons
Chain & clasp	• Pre-built mechanism; just need to attach the ends to bracelet ○ Multiple sizes	• Needs to be precise to hook onto the right chain link => takes more time to adjust • Dangling part of chain could be annoying (could tuck away)
Pulling both ends	• Simple knotting technique • Can be done with just string, so easier to find materials • Very fast/easy to adjust	• Now you have 2 long ends dangling, depending on how far you pull them; the loose ends can be distracting ○ Can tuck away the ends, but that would have to be done twice (since the ends are in opposite directions)
Cord lock	• Pre-built mechanism ○ Multiple sizes • Very fast/easy to adjust	• Still has a long string dangling, but it's easier to tuck away since there's only one end to deal with

Figure 10: Fastener Results

Based on the pros/cons, we chose the cord lock, because we value convenience in adjustability, as well as the ends not getting in the way. For different shapes of cord locks, we went with the circular shape because there's more surface area for the button press in comparison to the more narrow, rectangular ones. We also chose small ones to not be too bulky.

6 SYSTEM IMPLEMENTATION

6.1 Circuit

Fig. 11 shows what our first prototype of the system looked like. In this diagram, all components are connected to a common ground. An Arduino digital interface is used for the Beetle's connection to the LED array, and an Arduino analog interface is used for the connection between the microphones and the Beetle [11]. RGB values for the LEDs are calculated on the Beetle and sent to the LED array [5]; microphone input signals will be translated into decibel levels using a custom function that processes and interprets the microphone signals.

For the final version, we added peak detector circuit between the mic output and where it is read by the Beetle. More detail about the peak detector system is in a later subsection (6.1.3). Since the button to switch between instantaneous and average mode is no longer necessary (see 6.4.1), so the pin that would receive an input from that button remains disconnected. We discovered this after testing the average function, which was after the button components were already included in the PCB, so for consistency it's also in the newer diagram (Fig. 12). Date Calc [22] and the Button tutorial [17] were used to program the previously included mode button.

6.1.1 Battery System

The battery system works on two levels. On one level, ground and the 3.7V output of the battery connect to the GND and BAT pins of the Beetle, respectively (with the on/off switch between BAT pin and the + side of the battery). As the Beetle is powered, it uses its 3V3 output to supply power to the microphones. On another level, the 3.7V battery output and ground travel to the boost converter, whose output is used to power the LEDs. The output of the boost converter plugs into the voltage-regulator pin on the PCB.

6.1.2 Microphone System

The microphone system contains microphones, decoupling capacitors, the microphone's gain resistor between its OUT and GAIN-SELECT pins, and the peak detector circuit (see 6.1.3).

The original version of this system did not include a peak detector circuit and was designed to use through-hole microphone gain resistors, so that values could be swapped out during testing because we didn't know what gain value would be best. Initially, the resistors were bent to hold them in place on the board, but that proved insufficient to maintain a good connection. Electrical tape was then used to hold them in place, but it was messy to clean up and still didn't provide a stable enough connection. Further examples of testing resistors soldered them directly to the board, which was slightly harder to replace (if needed, we could still solder more resistors in parallel or series, but that required more finesse in soldering ability) but was much more likely to create a good connection.

After testing the extreme ends of the recommended range (0 Ω to 33k Ω), we found that none of those values were very responsive to sound, although 33k did the best comparatively. This made us hypothesize if the trend of higher sensitivity continued to higher values of resistors, so we tested values outside of the range, such as 51k Ω, 100k Ω, and 200k Ω. Of those values, 100k Ω performed the best in terms of sensitivity in picking up sounds, which is why we settled on that value for the final product. It should be noted that 100k is an unconventional choice for mic gain resistance because it is significantly far from the highest listed value on the microphone datasheet [3].

Before realizing it was a connection issue, we weren't seeing much difference in microphone outputs with the different resistor values, so we decided to go with 10k Ω since

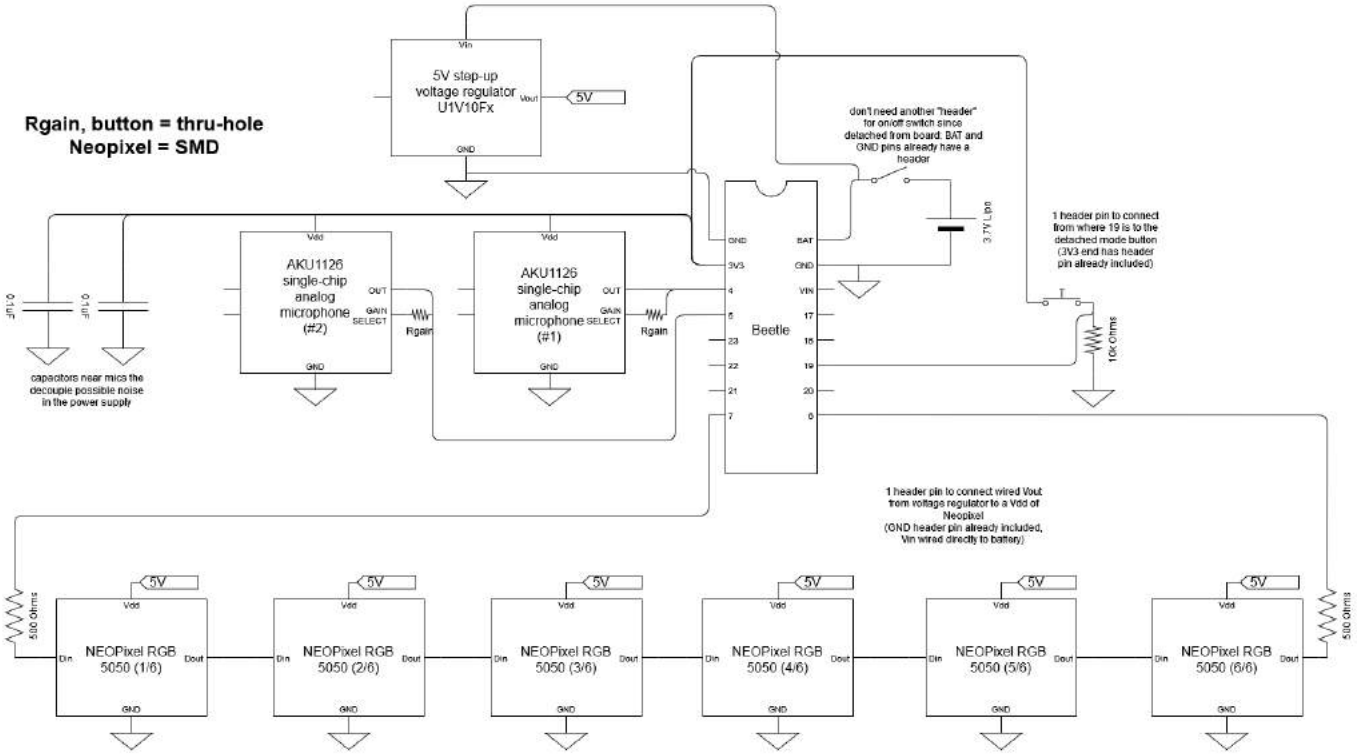

Figure 11: First prototype circuit diagram

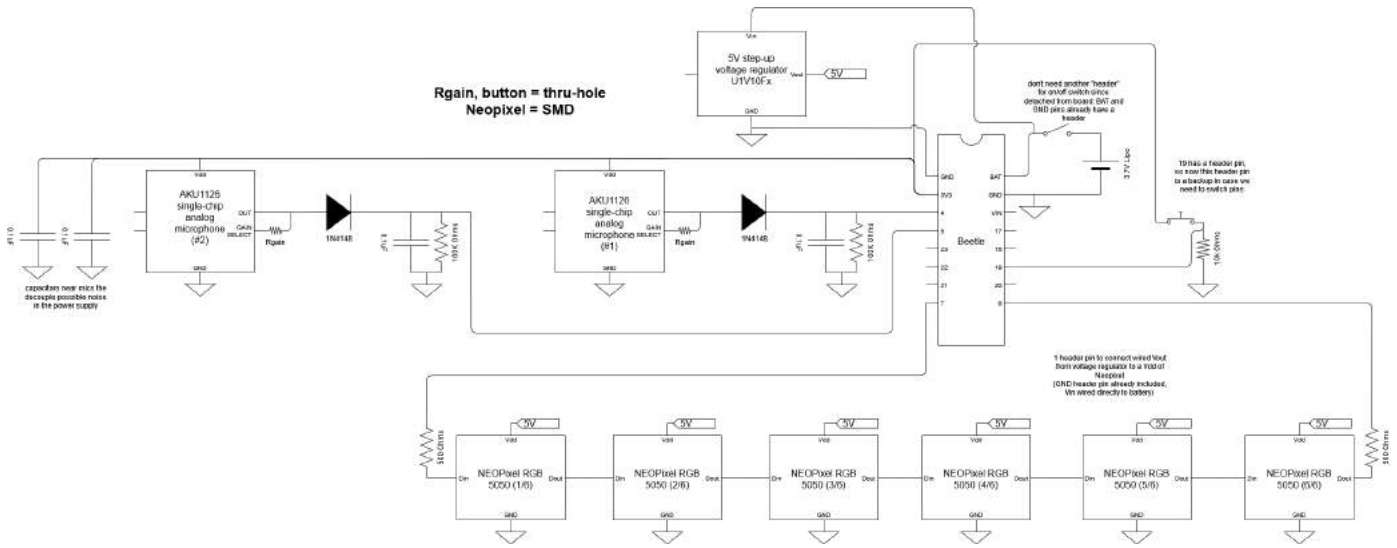

Figure 12: Intended “final” circuit diagram

it was in the middle of the range, and was in the PCB lab as a SMD resistor already to save on additional soldering. But after we soldered them properly and started testing, we had some worries of whether the best resistor value that we would end up finding would be available in the PCB lab as SMD, since we had already placed the order. Thus, we tried strategies such as stacking SMD resistors in parallel to achieve the correct value, or soldering the through-hole resistor legs to the SMD pads, both of which worked. Luckily, the resistor value we settled on was available in the PCB lab already, but we had enough backup options to go off of as well in case that didn't work.

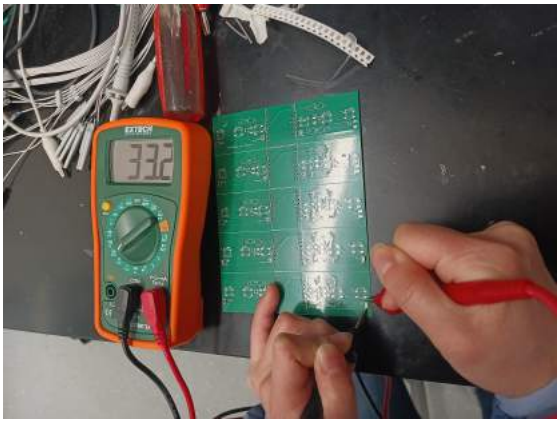

Figure 13: We stacked 3 100k Ω resistors in parallel, and successfully got about 33k Ω as desired.

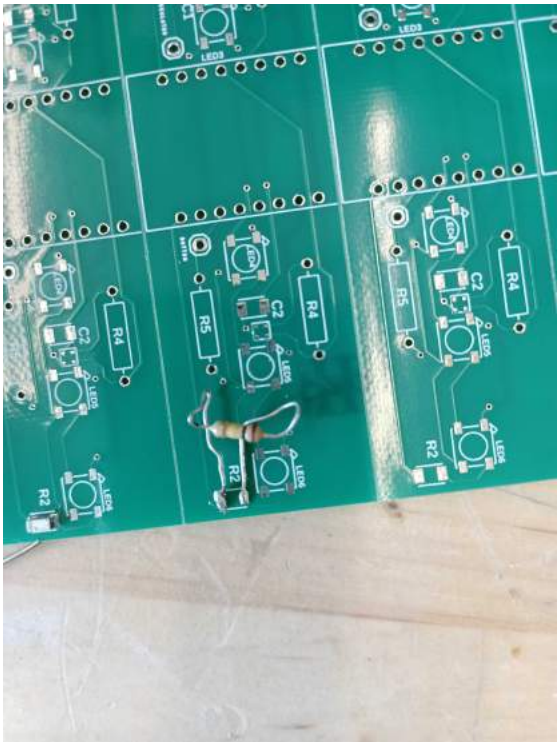

Figure 14: Soldering a through-hole resistor to a SMD pad also works, as checked with a multimeter.

Previous iterations of microphone testing code used PySerial [19] to print serial monitor values to a CSV file that could be graphed; an easier-to-use and preferable option is Arduino's built-in Serial Plotter.

6.1.3 Peak Detector

The peak detector circuit consists of a switching diode in series with a resistor and capacitor in parallel to each other (Fig. 15). (Before finding a source of switching diodes, we had considered using a rectifier diode because of its ready availability.) The voltage output of the peak detector circuit is smoother than the input, because the capacitor's collecting charge allows the system to react to surrounding events by parsing sound waves into more consistent lines. This means that the microcontroller can use a lower sampling rate so we wouldn't run into Nyquist frequency sampling problems (though this is no longer an issue as analogContinuousRead allows high sampling frequencies [24]), but if we were instead using analogRead at fixed intervals [6], the peak detector would render the circuit output more stable and make it more likely that the microcontroller would detect something related to a peak of the output rather than a valley where nothing happens.

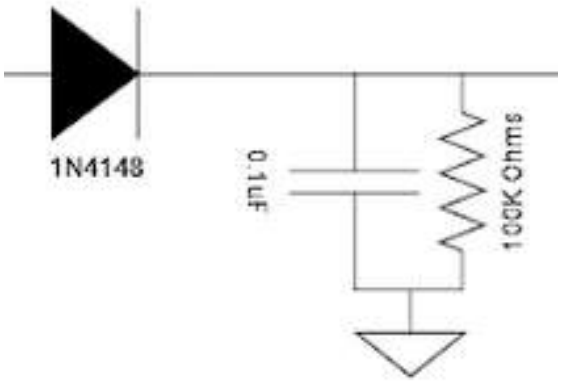

Figure 15: Peak detector circuit. The microphone output goes into the diode, and the other side is read by the Beetle.

6.1.4 LEDs

The LEDs connect in a series, each of which has pins for DATA-IN and DATA-OUT, as well as for power and ground [21]. The input and output into the first and last LED have a resistor in the range of approximately 500 ohms, as recommended in the datasheet [21]. We ended up using a 470 Ω resistor, since that was the closest SMD resistor available in the PCB lab.

6.2 PCB Design

The PCB was designed using Autodesk Fusion, which is free with a student account. First, you create a circuit schematic, which will inform the program what components

will show up in the PCB layout file. Usually, one can drag-and-drop components from the many provided libraries, but unfortunately, the Beetle and microphone brand were rare enough to not be included. Thus, we had to custom-design those components and add them to our "Custom Library" first. This was a new experience, so YouTube tutorials were very helpful here [13]. Below are the pictures of the microphone and Beetle from the Custom Library, in Figures 16, 18, 17, and 19. See Fig. 20 for the completed circuit.

Figure 16: AKU1126 Microphone schematic

Figure 17: Beetle schematic

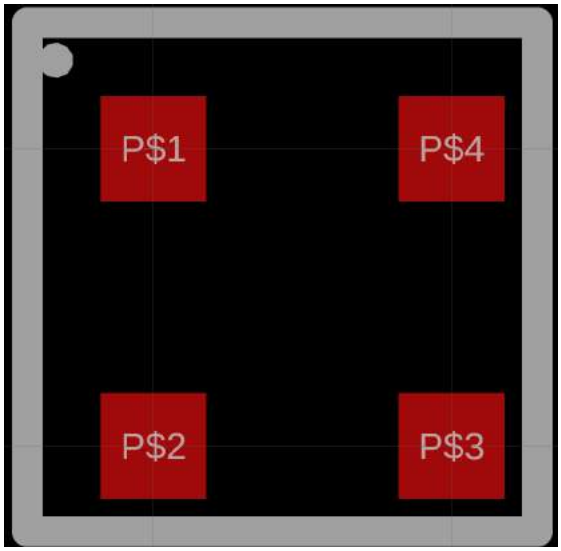

Figure 18: AKU1126 Microphone PCB footprint

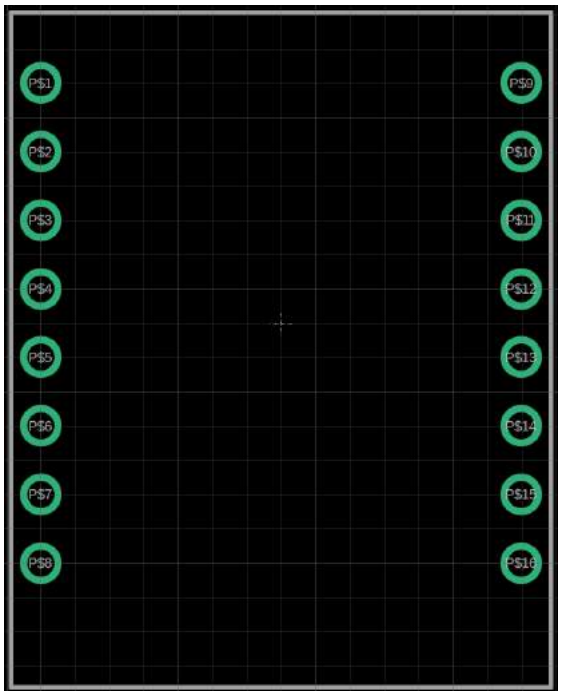

Figure 19: Beetle PCB footprint

Next was arranging all the components onto the PCB and drawing traces between them. There were certain components like the lights, microphones, and Beetle that had fixed locations, but after that it was mostly about arranging things to minimize trace widths, as well as drawing paths to not cross where they weren't supposed to. The traces for power and ground, as well as others like DATA-IN/OUT that had to cross the Beetle were thickened to decrease resistance. See Fig. 21 for the PCB for the first prototype.

For both the circuit schematic and PCB, the last step is to click on the ERC button, which does some basic rules checking to make sure all your connections are expected.

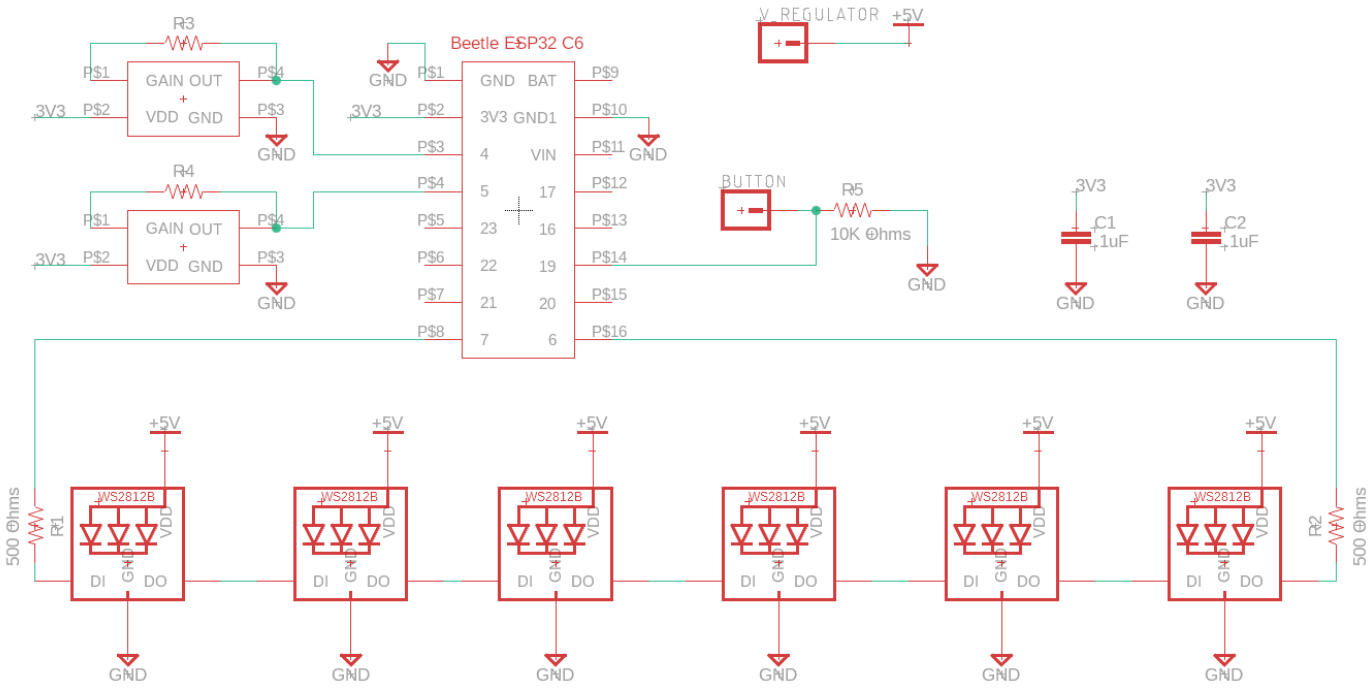

Figure 20: First circuit schematic prototype of the PCB

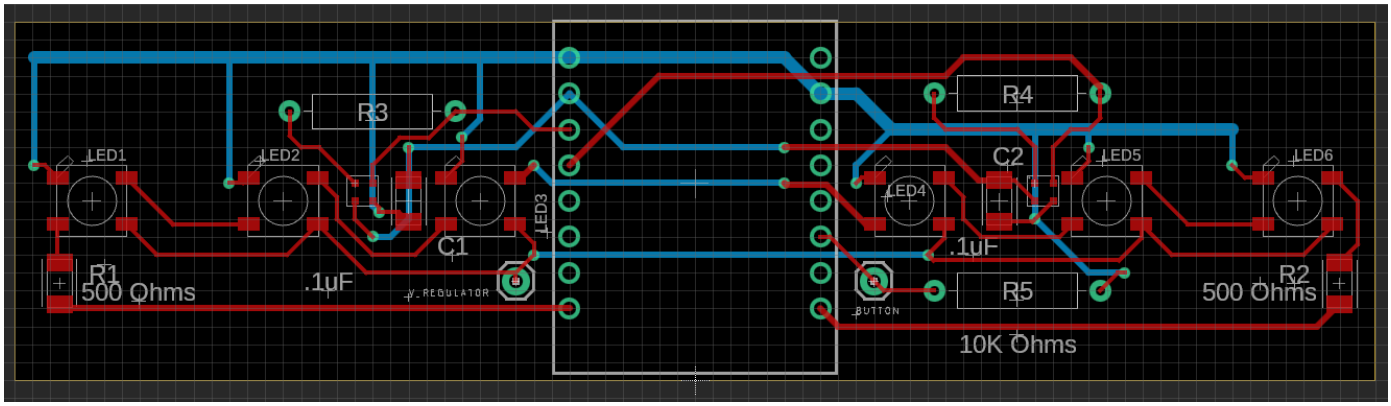

Figure 21: First PCB layout prototype

As mentioned above in the Circuit section (6.1), we ended up adding a peak detector to the microphone outputs, which required altering the circuit schematic and PCB layout. Luckily, the diode was already in one of the provided libraries, so we didn't need to go through the process of creating a new part. Unfortunately, updating the circuit schematic didn't automatically update the components provided in the PCB tab; instead the program would just complain there were inconsistencies. So that meant making the PCB layout from scratch, which was mostly fine except for when the components wouldn't snap to the same places that it used to in the previous design, despite the same grid granularity settings. Google didn't return any helpful results, but this was fixed by shuffling components and then moving them back. Finally, the through-hole resistors for the microphone gain values or the button became SMD for ease of soldering. The new circuit schematic and PCB layout are provided in Figures 22 and 23.

Submitting the design to be created required downloading a zip file containing the Gerber file from Autodesk Fusion and uploading it to the design submission space. If correctly uploaded, the order submission document would use the exact dimensions specified in the Gerber file.

The following are a selection of settings used when ordering the PCB. The minimum thickness was selected for the rigid board in order to decrease its bulk; this was useful when certain boards had to be bent slightly in order for the light connections to work. Conversely, when ordering the flex PCB, the maximum thickness was chosen so that the board had the best chance of standing up to through-hole soldering and for stability. Untented via coverings were chosen for the rigid boards, for easy soldering [1]. A framework stencil was chosen instead of a step stencil because only one level of solder was necessary [1].

To assemble the PCB, first line up the stencil with the boards SMD pads and tape it in place (this took a while for us since the holes were so small it took a lot of tiny adjustments). Then, apply solder paste over the stencil, evenly coating the exposed metal. Untape the board (you can remove the solder paste from the stencil with rubbing alcohol). If the soldering is mostly fine except for a few small areas, fix that with tweezers. Otherwise it's just easier to redo the whole thing. Use tweezers to pick and place the components. We recommend starting with the smallest components (mics) first, because you have the most room to maneuver it into place without being blocked by another component. For the mics and LEDs, pay attention to the orientation because it matters for the data connections (unlike for resistors and capacitors).

While you were placing components would've been a good time to start preheating the oven: just follow the steps that are helpfully taped on there. The oven will beep when done. Set the board on the tray, and if the board light/small, it's probably a good idea to secure it in place using wires or other heat-resistant material.

After the oven is done baking, you should be careful when taking the board out because it will be hot: wait for it to cool and/or use tools to avoid directly touching.

For hand-soldering, you can do the diodes first when you have a somewhat flat surface. To make sure the header pins are aligned nicely, it helps to use some tape to help hold their position. Don't press too hard on the board otherwise you'll shift them. Then, solder one of the pins on each side of the row of header pins. Now they'll both be secure and won't wobble as you solder the rest. Repeat with the lone headers.

For the flex PCB, apply a blob of hot glue to the underside of the single header pins to help stiffen the area and prevent them from being ripped out.

6.3 Casing

There were several casing designs considered and prototyped before we found one that worked decently well.

6.3.1 Plastic Tubing (V0)

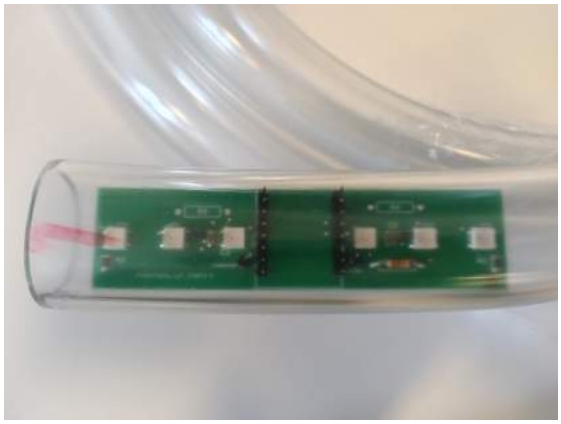

Figure 24: Hard PCB prototype in plastic tubing

First, we tried the plastic tubing, because that would minimize the amount of gluing needed in comparison to cutting from a flat piece of plastic. Unfortunately, the tubing walls were pretty thick compared to the plastic sheet, so that would add a lot of height to our product (see Fig 25). It also wasn't wide enough to fit the battery along with the PCB, so we decided to not use it moving forward and switched to the PVC Vinyl sheet instead.

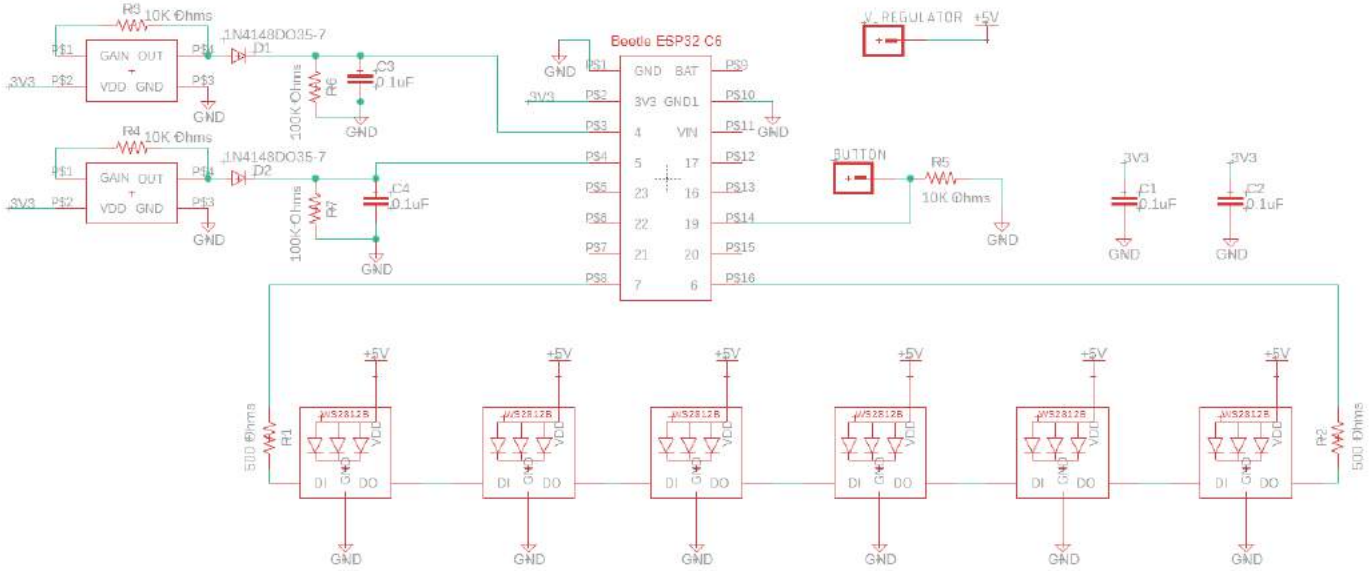

Figure 22: Final PCB circuit schematic

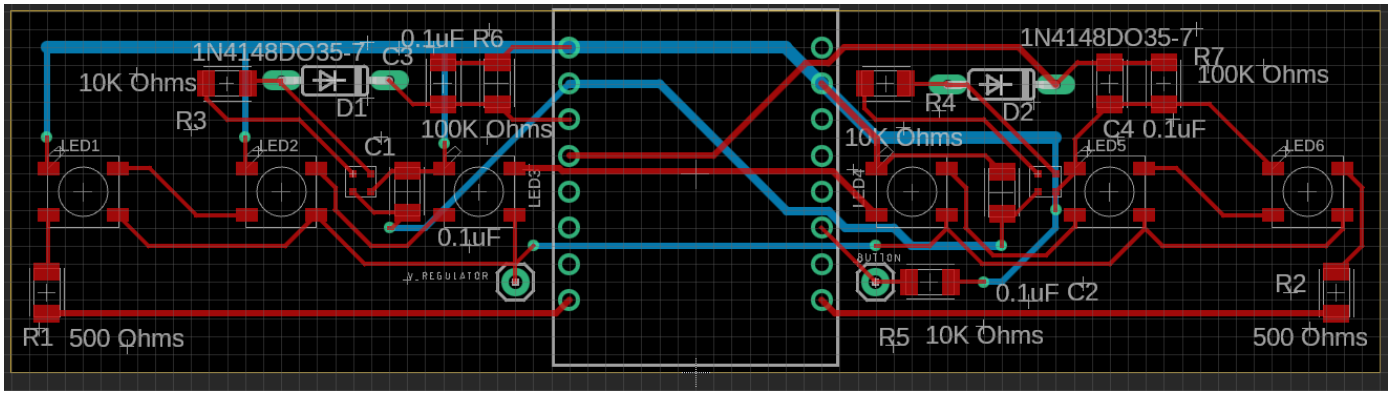

Figure 23: Final PCB layout

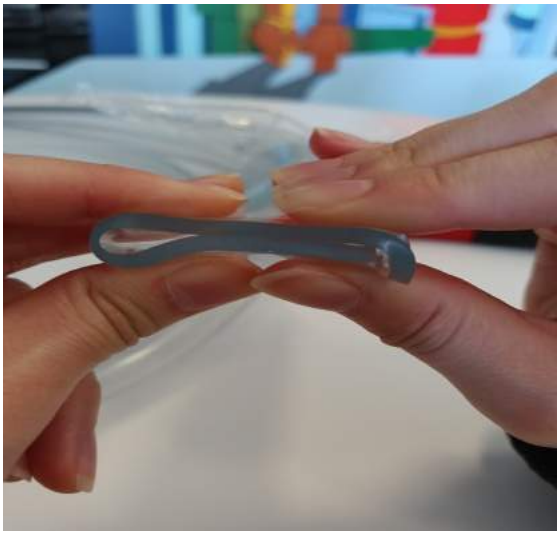

Figure 25: End of the tube

6.3.2 Rectangle Cut (V1)

For the plastic sheet, the first thought was to cut out 2 rectangles with some buffer room to seal the edges. This iteration used 1cm buffer for the sides, resulting in rectangles that were about 10.5cm x 5cm.

Figure 26: Rectangle Top View

The edges were sealed with a hot glue gun, except for the place where the USB charging cable would go. Unfortunately, it was not very susceptible to bending because of the small surface area that the glue covered. This gave the conclusion that the next iteration would need more overlapping surface area in order to stay in 1 piece. During this iteration, we also learned how to use a metal hole puncher to cut the holes for the mics: this was much easier than trying to stab the plastic with a sharp object, since it was surprisingly stab-proof.

6.3.3 Trapezoid Cut (V2)

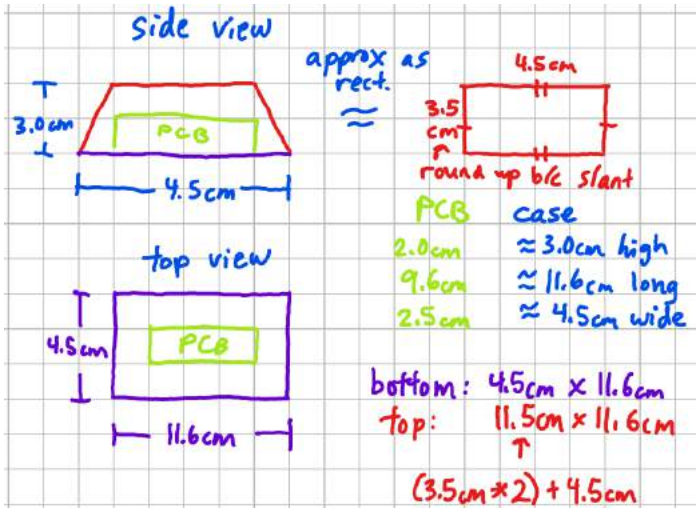

Figure 27: Trapezoid Calculations

For the next iteration, the top and bottom pieces would be different sizes in order to have some extra room to help overlap the edges. This resulted in a "trapezoid" approximation of the case. The calculations for the dimensions are shown below, in Fig. 27. The good news was that the edges stayed sealed pretty well this time. The bad news was that the overlapping of edges from both sides made this a bit bulkier than needed (see Fig. 28), as well as making construction more difficult.

Figure 28: Trapezoid Side View

You'll also notice that the "trapezoid" approximation was not very accurate, since it likes to retain the circular shape that the wrap came as. Thus, for the next iteration, we'll try approximating as a circle.

In the meantime, a next step is to try ways of sealing the ends. Just pressing the edges down wasn't going to work, as seen in previous iterations. Thus, a more clever interlacing method was used to give more surface area and to help lock things together (see Fig. 29). Basically, the edges now have tabs cut instead of being a flat surface. The tab alignment of each edge is offset such that the tabs

can fit nicely into each other's gaps and fold onto the other edge to be glued down. Fig. 30 shows the result.

While it works pretty firmly, it requires some precision to plan out the tab spacing and offsets, and requires some extra space on the edge to cut into tabs, because tabs that are too small won't have enough surface area to grip onto the other side. The final method of fastening the edges will be explained in V3 below.

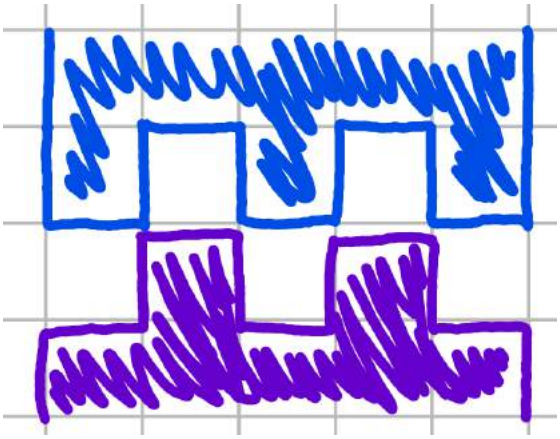

Figure 29: How edges will fold/lace together

Figure 30: Laced edges

6.3.4 Circle Cut (V3)

Since the battery width is larger than the PCB, that was used to determine the minimum diameter of the circle (the height of the battery + PCB were not as large as the width of the battery). The dimensions were found from the specs sheet, and then rounded up to have some extra circumference left over for overlapping the edges. The case was cut as a rectangle, where one side would be the length of the PCB + 1cm buffer on either side (same as before), while the other side of the rectangle would be 4πcm, which rounds to about 12.56cm.

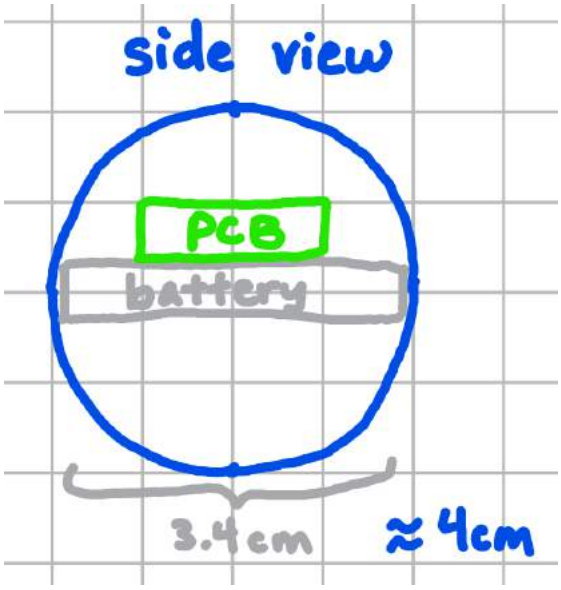

Figure 31: Circle Calculations

As with the previous iterations, holes were also added to accommodate the microphones, USB cable, and switch.

The 1cm buffer room wasn't enough to effectively lace the ends together as with the previous iteration, so instead, the ends were held shut temporarily with glue while a rectangular strip of plastic was applied perpendicularly to the edge so it bridges both sides. The polyester cord was inserted into the gap between "fasteners", and also hot glued for extra stability. See Fig. 32 below for final result.

Figure 32: Fasteners

6.4 Software

6.4.1 Bracelet Code

The bracelet code combines code for LED control and microphone system input responses. The LED control code sets each LED's color value depending on if the microphone reading reaches a certain threshold value. The microphone code begins by taking the average of many trials; it uses this value as a baseline to which it can compare subsequent voltage values entering from the analog connection. This is because we discovered that the microphone baseline varies greatly each time it's plugged/unplugged, so basing thresholds on a constant would not be accurate. However, the distance that the voltages span to reach the threshold do stay similar, so we can calculate our thresholds based on the starting values.

Once a baseline is established, analogContinuousRead [24] is used to get values from the analog input connected to the microphone system and compare them to the threshold values. If a threshold is reached, the LEDs change color accordingly. Threshold values were calculated by using WolframAlpha to find the logarithmic line of best fit between other microphone gain resistor values and our resistor value [30], using the microphone sensitivity converter for converting our resistor's decibel level to mV/Pa [9], then using the sound pressure level converter to obtain a Pa reference for 80 and 100 dB [10]. This allows the thresholds for 80 and 100 dB to be directly calculated as a constant value. In our case, the 80 dB number would be 149, and 100 dB would be 1490. These values are added onto the baseline average calculated previously to determine the mV value that the microphone needs to reach in order to change color.

A previous idea was to try a linear representation of the microphone values to calibrate between mV and dB, or use other types of curves such as exponential or power functions. However, the best fit coefficients we found would always be dependent on the baseline average, which isn't constant. Our new method of using the mV/Pa conversions doesn't depend on the specific startup value, making it more dependable.

One dead end involved implementing average mode. Average mode initially updated the average with every reading, starting with the beginning of a session. However, there were a few problems with this. Firstly, this could lead to overflow in the code if it ran for a long enough time, Secondly, fast loud noises would not influence the average enough because the majority of the readings would remain below the threshold. An updated average-mode calculation used the Arduino library runningAverage [25] to only average a constant number of the most recent samples. While this solved the overflow error, it showed that average mode was not a reasonable goal for our system. If a large window of samples was chosen, then it would have the same issue as having an "infinite" window (taking into account everything from the beginning of the session): peaks above the threshold would not affect the total average enough. If a small window was chosen, then the result became closer and closer to instantaneous mode, and still would have the issue where averaging would land below a threshold even if values peak above it. For those reasons, average mode was not included in the final version of our system.

6.4.2 Web App Development

The web application was made using React Native and Expo, a popular software framework that uses JavaScript to build cross-platform apps. The first screen of the app is an authentication page, where the user is prompted to either log in to an existing account or sign up. Afterwards, the user is brought to a home page, where they can navigate across different tabs. In the settings page, the user can choose to connect their bracelet to Bluetooth, as well as customize the light intensity, threshold levels, and color of the bracelet. At first, our app was displayed using a QR code that brought the user to another app called Expo Go. This app would bundle all the code and display the UI. However, Expo Go unfortunately does not support Bluetooth libraries, so we decided to switch over to a developmental build. This type of build allows for app development in either Xcode, Apple's integrated development environment for macOS, or Android Studio, the integrated development environment for Android systems. These platforms also have phone simulators that can display the app, but unfortunately also do not support Bluetooth. Therefore, our only option for our app was to generate it on a physical phone. We decided to move forward with Xcode and iOS after running into package errors with Android Studio.

6.4.3 Bluetooth

For the Bluetooth connection, there were two external react native libraries that support BLE: react-native-ble-plx and react-native-ble-manager. At first, we went with ble-plx because it was the most popular library and had good documentation [14]. We ran into issues with ble-plx, since this library only supported up to Expo SDK 49, while the rest of the app was running on SDK 50. This made it extremely difficult to integrate with some of this library's features. We switched to ble-manager after trying and failing to make a workaround with ble-plx. However, ble-manager kept crashing the app, which caused us to switch back over to ble-plx. Unfortunately, due to all of these setbacks, we ran out of time to fully integrate Bluetooth into our system.

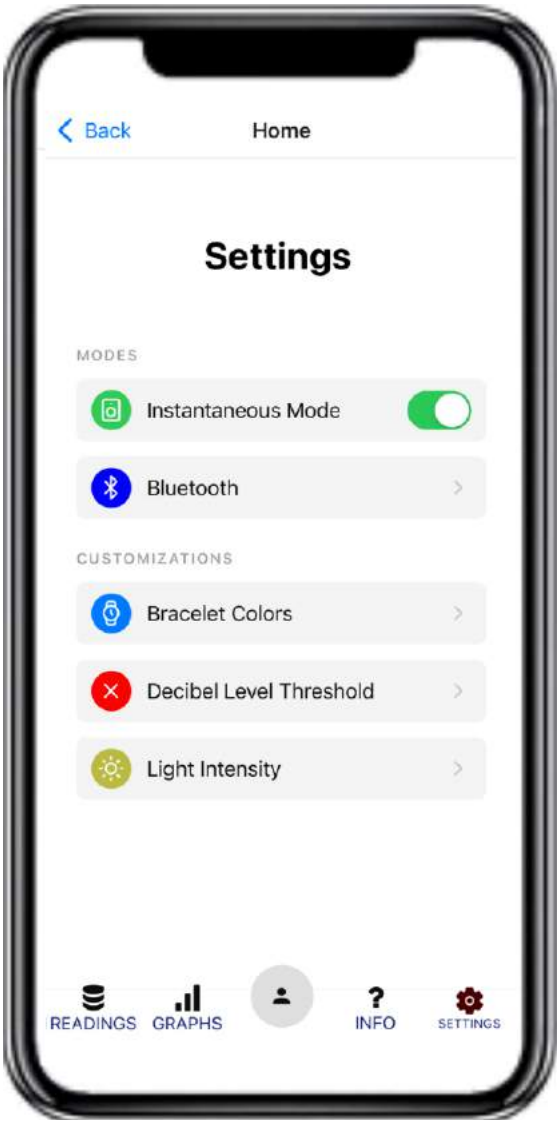

Figure 33: Settings Screen

7 TEST & VALIDATION

7.1 Results for Weight

Figure 34: Bracelet Weight

As shown in Fig. 34, the bracelet weighs less than 200 grams, so the weight test is passed.

7.2 Results for Width and Thickness

Figure 36: Bracelet Width, Side 2

Figure 37: Bracelet Thickness

As described in our design report, width and thickness were tested with a straightedge for measurement. The width on either side was slightly larger than 46 mm (≈ 60mm at the widest point), and the thickness was also slightly greater than 12 mm (≈ 30mm); one possible reason is that we are using the 2000 mAh battery, which is slightly larger than the intended 850 mAh battery (which never arrived). Another reason is that the internal connections in the bracelet are created using jumper cables secured with electrical tape; soldering wires cut to the correct length would have allowed for more efficient filling of space within the bracelet.

7.3 Results for Operating Temperature

The planned test for operating temperature has been replaced, since we do not have access to a temperature sensor. While in use, the case does not feel more than slightly warm when touched; therefore, it is fair to assume that it does not reach 105 °F. For extreme versions of the test, the bracelet was left operating for over 4 hours, and still is only slightly warm to the touch, in that someone could be comfortable touching it indefinitely without being burned. This version of the test passes.

7.4 Results for Accuracy and Timeliness

The planned test for accuracy has been replaced by a new test. We did not use a soundproof room or connect to the bracelet. Our test setup was playing loud music at the bracelet and looking at a) the serial plotter's response on Arduino (graphing inputs from the peak detector) and b) the LED colors. Relative angle was not tested, since we are no longer pursuing directionality; the optimal angle for microphone sound pickup was found via experimentation and preferentially used after that.

Accuracy does not always reach the requirement of being within 2 dB of the sound meter's reading: sometimes, the sound meter reads higher than 82dB (accounting for the 2dB leeway) but the microphones don't reach the threshold, although they do get close. However, this is also due to a flaw in our testing requirements, where we weren't detailed enough in our specifications for accuracy. In particular, it would've been helpful to include accuracy as a percentage for false negative/positive rates, as the other teams do, because most systems aren't 100% accurate either. For example, false negatives for us include microphones not being sensitive enough or the Beetle entering sleep mode and no longer responding to sound. False positives include poor connections leading to the bracelet changing to the wrong color.

Timeliness passes in terms of the lights changing color very quickly: the response time is less than half a second from our video evidence. App responsiveness could not be tested.

Another thing to note is that the sound for these tests did not approach 100 dB, because we don't have speakers (or hearing protection) loud enough. So it is possible that the danger threshold is inaccurate. However, we think the threshold set is reasonable given that the 80dB threshold we calculated the same way is somewhat accurate. For this test, the sound meter used fast response and A-weighted settings (A is the standard weighting [26]), and we generally tried to set it to high (the setting for loud environments). Our test music tended to include high frequencies, because the microphones are more sensitive to higher frequencies according to the datasheet.

7.5 Results for Adjustability

While the bracelet cord looks short, measuring the circumference reveals it to have a range of 9-11 inches (228.6-279.4mm). This passes our maximum end of the requirement of 10 inches. However, its minimum size is bigger than the intended 7 inches. This is because while the flex PCB and plastic are both pretty bendable, the case has been stuffed with stiffer components like wires and battery, which limits the entire flexibility to not bend much at all. Thus, the minimum circumference of the PCB became determined by being about twice the length of the PCB, instead of being less than that.

7.6 Results for Durability

The drop test was performed as described in the design report (besides not testing Bluetooth), where it was released from rest at 2.5ft off the ground. The test happened three times. The first time, the bracelet required a reset to return to previous functioning; the next 2 times, it did not. (Since it is difficult to verify microphone functioning as normal with our setup of the test, the results were determined based on LED colors.) In further testing, after

being dropped, the bracelet was able to display values on the serial plotter when plugged in.

7.7 Results for Battery Life and Comfort

The planned test for comfort was replaced by two team members trying on the bracelet; both could get it to fit, but the response was that the bracelet seemed bulky, although decently light.
Battery life was tested in two ways. One way measured the current drawn from the battery when the bracelet system is in use (mics taking in sound data, LEDs lighting up). The maximum current drawn was 60 mA. In theory, with yellow lights (the color that seems to have the largest current draw out of our default colors), it would take 31 hours for the battery to drain to zero if the brightness is set to 50 out of 255; if set to maximum brightness, it would take 12 hours. Another test required setting up the bracelet system (without casing) and playing random opera music at it for 4 hours, while checking periodically to see if it has run out of battery (or entered deep sleep mode, in which case a reset would be necessary). After 4 hours, the bracelet system still changed color in response to high-frequency and loud sounds, confirming that its battery was still functioning.

7.8 Results for Web Application

We could not complete the integration tests between the web app and bracelet since we couldn't establish a connection.

UI user testing was carried out. About 5 people evaluated the web app and rated its ease of navigation as on a scale of 1 to 5, where 1 is awful and 5 is amazing. The average rating was approximately 4 out of 5. The majority of opinions believed the web app was well designed and easy to navigate. There were small areas that could be approved on, including the location of the logout button and the way the information page was set up.

For security, we created an authentication system where only users who have logged into their own account can access the app. This was tested by creating 10 different accounts, each with a different email address and password combination. Only when the correct email and password were inputted did the app bring the user to the rest of the app, where they can access their personalized profile. Each profile page was updated with the email address of the current user in the app. Testing storing and outputting data into graphs did not happen; neither did confirming the accuracy of adjustable settings.

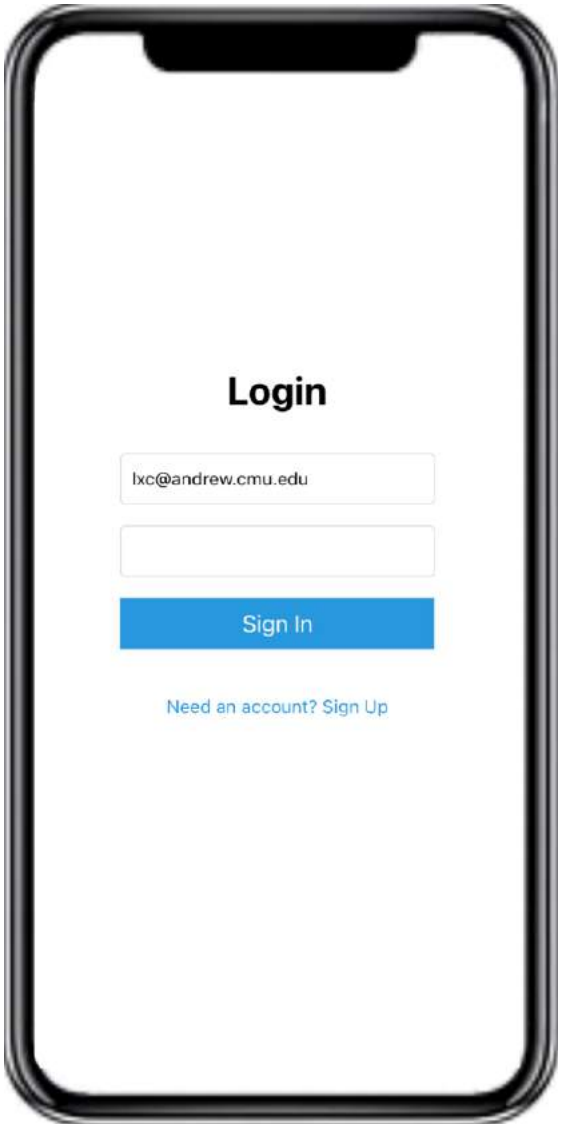

Figure 38: Login Screen

8 PROJECT MANAGEMENT

8.1 Schedule

Our updated schedule is shown in Fig. 39.

8.2 Team Member Responsibilities

Lucy Chen was primarily responsible for the creation of the web application UI and design. She coded both the frontend and backend of the app. She was also responsible for the Bluetooth connection on the web app side of the project. Additionally, she assisted in the hardware side by assembling PCBs.

Katherine Sabak was primarily responsible for the signal processing and hardware of the bracelet. This mostly entailed signal processing and testing different electronic combinations to optimize the microphone signals. She also assisted in checking the PCB design, writing integration

Table 1: Bill of materials

Description	Model #	Manufacturer	Quantity	Cost @	Total
Beetle ESP-32 C6	DFR1117	DFRobot	1	$4.90	$4.90
NeoPixel RGB 5050 LED w/ Int. Driver Chip (100)	3094	Adafruit	1	$29.95	$29.95
Pololu 5V Step-Up Voltage Regulator	2564	Pololu	1	$5.95	$5.95
BATTERY LITH-ION 3.7V 2AH	1528-1857-ND	Adafruit	1	$12.50	$12.50
INSPIRELLE 3mm BlackSatinCord, 50 yd spool	-	Inspirelle	1	$9.99	$9.99
Deep Dream 1.5mm Clear Desk Cover Protector	-	Deep Dream	1	$9.99	$9.99
Adj. Cord Lock Round Ball Sing. Hole End Tog.	2135-4001	SpecialistID	1	$0.19	$0.19
PCB w/ solder stencil	-	JLCPCB	1	$47.72	$47.72
Button, Switch	-	IDeATe lab	1	$0.00	$0.00
AKU1126 Single-Chip Analog Microphone	AKU1126	Akustica	2	$0.00	$0.00
SMD resistors and capacitors	-	PCB Lab	4	$0.00	$0.00
Solder paste and oven	-	PCB Lab	8	$0.00	$0.00
Peak detector diode	-	IDeATe lab	8	$0.00	$0.00
Hot glue, hole punch, soldering materials	-	Campus Makerspaces	-	$0.00	$0.00
					$121.19

code, ordering parts, and helping with the Bluetooth connection.

Freda Su was primarily responsible for the physical design of the bracelet, which included manufacturing the bracelet casing and setting up the LED signals. She also worked on PCB design, fabrication, and assembly, as well as writing the Arduino code to integrate the bracelet system. In addition, she helped with testing and calibrating the microphone signals with Katherine.

8.3 Bill of Materials and Budget

Materials that we bought and did not use for the final product include the 850 mAh batteries, the first plastic wrap (both previous examples never arrived), tubing from McMaster-Carr, and PCBs and a stencil for the first PCB design. Materials that we did not include in our design report but needed to complete the system include a breakout board for the battery output connection (shared by a friend).

8.4 Risk Management

8.4.1 Design

One of the greatest risks that our device suffers from is fragility. We successfully mitigated this risk by using a plastic casing to cover the PCB and battery, so that the entire bracelet will have a coating protecting both the user (electrical insulation) and the components. Multiple versions of the casing were prototyped to find the best design (6.3).

Another risk we had in development was with the PCB assembly in that some components were harder to solder properly, such as the mics. Therefore, since we had an ample budget, we ordered many spare components and built over a dozen PCB boards to increase the chance that at least one of the boards would work. First, we practiced soldering with rigid PCBs because they were much easier to work with and to discover preliminary issues, which we would then fix. Then we transitioned to using the much more fragile flex PCBs.

8.4.2 Schedule

The greatest scheduling-related challenge is running out of time. We built in buffer time at the start of the semester, but ended up working through the end of the project timeline. One way to mitigate this issue is to work on subsystems concurrently, but some take longer than others, so we can still end up getting blocked. A solution is for teammates who are more free to help out with other teammates' tasks.

8.4.3 Resources

The greatest resource-related challenge is being unable to receive ordered parts in time (or at all). A mitigation strategy is to use resources on campus where free parts are available, such as Roboclub, TechSpark, and IDeATe. We also improvised with the resources we had, such as with producing new values for the microphone's gain resistors, as mentioned in 6.1.2 for details.

Another resource-related issue is the limited number of opportunities we had to test our device in the music studio. A strategy for getting as much data as we can is to coordinate with the music students early in the hopes of finding mutual free time.

9 ETHICAL ISSUES

There are some ethical issues that are related to our project. We wanted to create a bracelet that was durable, economic, and safe. However, since our project is made primarily of electronic components, they are not the most environmentally friendly. This informed our decision making process by requiring the bracelet to be strong and durable.

Therefore, it can be long lasting and not need to be replaced often, reducing the amount of waste of our product.

Next, our bracelet is meant to be worn for a significant duration of time, meaning a device malfunction could potentially cause harm to the user. We helped mitigate this by wrapping the bracelet in a plastic wrap to prevent any electrical components from touching the user. Additionally, since the bracelet is meant to track noise levels, a malfunction could cause the product to display lights that inaccurately reflect the user's current surroundings. This could cause the user to falsely assume their noise levels are safe. Also, since the microphones are sensitive to distance, another concern with accuracy is that the mic readings would be significantly different an arms length distance from the ear, making them read in values that are some decibels lower than the noise at ear level.

Also, our product is, at its core, a sight-based tool for checking noise levels. Therefore, our bracelet does make it difficult for people with vision disabilities to use the product. However, we did try to make it accessible by adding a feature, color customization, that allows for colorblind people to customize bracelet colors to choose colors that work best with their vision. Our product does unfortunately exclude fully blind people as the bracelet is designed with sight in mind.

Furthermore, privacy could be a concern, as the web app component is designed to collect and store personal data about the user. We tried to mitigate this with the use of password authentication, where the app only works if the user has logged in to their account.

10 RELATED WORK

The closest related work to the system we have achieved is CyberJewelry, which was a previous capstone project [7].

11 SUMMARY

In summary, our system was able to meet some of the design specifications. Microphone sensitivity and accuracy for processing sound inputs may have been limited by microphone properties or connection issues on the PCB. There are some relatively simple fixes that would improve the performance of our system. Another is redoing the PCB-battery system placement into the casing so that all connections remain stable: instead of using standard-length jumper wires and electrical tape, soldering in wires that have been measured for the optimal length would improve the internal connections by leaving more room within the bracelet.

11.1 Lessons Learned

Working on this project has taught us several valuable lessons, which we can now share.

First, make sure all your electronic connections are stable using soldering. Even though the metal components look like they're in contact, it still might not be a good enough connection to be stable. We added sockets to the Beetle to connect to header pins on the board in order to be able to swap boards if some parts weren't working without needing to find a new Beetle/desolder.

Miniaturization is non-trivial, both in finding components that fit specifications and in handling them. Small components are easy to break or lose, and difficult to pick up and place precisely. For components such as the mic where the soldering pads are small and underneath the plastic, it's hard to tell whether the solder is too much or too little (or just right) until testing after the PCB is done soldering, and by then we're not going to be able to redo it.

Tests need to be very precise, to the level of writing a step-by-step procedure. A standard procedure both reduces confusion about what counts as a pass or fail and makes setting up and repeating tests easier. Less obviously, having detailed test procedures written in advance gives members more time to look things over and discuss potential flaws in how the test will be conducted, so they can be revised to be better.

Glossary of Acronyms

- BLE – Bluetooth Low-Energy
- LED – Light-Emitting Diode
- PCB – Printed Circuit Board

12 Acknowledgments

We would like to thank two friends who provided debugging help with the webapp code, and helpful TechSpark employees (who loaned a phone for filming microphone outputs, advised on the correct soldering temperature, and explained how to use the metal hole puncher), as well as a friend who helped with microphone signal checking. Finally, thanks to Prof. Fedder and TA Zeynep for their suggestions on microphone debugging, as well as techniques for checking components such as with the multimeter and oscilloscope.

References

[1] URL: https://cart.jlcpcb.com/quote.

[2] *A-weighting*. 2024. URL: https://en.wikipedia.org/wiki/A-weighting.

[3] Akustica. "AKU1126 Single-Chip Analog Microphone and AKU2002B Digital Output Microphone". In: (2009).

[4] GBD 2019 USA Hearing Loss Collaborators et al. "Hearing Loss Prevalence, Years Lived With Disability, and Hearing Aid Use in the United States From 1990 to 2019: Findings From the Global Burden of Disease Study". In: *Ear and Hearing The Official Journal of the American Auditory Society* 45.1 (Sept. 2023), pp. 257 –267. URL: https://journals.lww.com/ear-hearing/fulltext/2024/01000/hearing_loss_prevalence,_years_lived_with.24.aspx.

[5] Phil Burgess et al. *Adafruit NeoPixel Library*. 2024. URL: https://github.com/adafruit/Adafruit_NeoPixel/tree/master/examples.

[6] *analogRead()*. 2024. URL: https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/.

[7] Madi Davis Shize Che Carnegie Mellon ECE Capstone Spring 23 – Saniya Singh. *Team A2: CyberJewelry*. 2023. URL: http://course.ece.cmu.edu/~ece500/projects/s23-teama2/.

[8] Brent Cohen. *How Much Does A Smartphone Weigh?* 2022. URL: https://devicetests.com/how-much-does-a-smartphone-weigh.

[9] ANVICA Software Development. *Microphone Sensitivity Converter*. 2024. URL: https://www.translatorscafe.com/unit-converter/en-US/microphone-sensitivity/.

[10] ANVICA Software Development. *Sound Pressure Level (SPL) Converter*. 2024. URL: https://www.translatorscafe.com/unit-converter/en-US/sound-pressure-level/.

[11] DFRobot. *SKU:DFR1117*. URL: https://wiki.dfrobot.com/SKU_DFR1117_Beetle_ESP32_C6.

[12] Center for Disease Control and Prevention. *So How Accurate Are These Smartphone Sound Measurement Apps?* 2014. URL: https://blogs.cdc.gov/niosh-science-blog/2014/04/09/sound-apps/.

[13] Will Donaldson. *Creating an Electronic Component Library in Fusion 360 [Part 1]*. 2022. URL: https://www.youtube.com/watch?v=NITJZfhjppI.

[14] Fimber Elemuwa. *"Comparing React Native BLE libraries"*. 2024. URL: https://blog.logrocket.com/comparing-react-native-ble-libraries/#:~:text=Unlike%20React%20Native%20BLE%20PLX,typical%20application%20needs%20reasonably%20well./.

[15] Rapport Furniture. *Standard Dining Table Height: How Tall Should It Be?* 2024. URL: https://rapportfurniture.com/blogs/rapport-furniture/standard-dining-table-dimensions.

[16] *Hearing Loss Decibel Levels*. 2024. URL: https://neworleansmusiciansclinic.org/health-clinic/health-topics/hearing-loss-decibel-levels/.

[17] *How to Wire and Program a Button*. 2022. URL: https://docs.arduino.cc/built-in-examples/digital/Button/.

[18] Bling Jewelry. *Find Your Bracelet Size or Bangle Size*. 2024. URL: https://www.blingjewelry.com/pages/bracelet-sizing.

[19] kevinjpower. *Capture Data from Arduino to CSV File Using PySerial*. URL: https://www.instructables.com/Capture-Data-From-Arduino-to-CSV-File-Using-PySeri/.

[20] Frank R. Lin. *The Hidden Risks of Hearing Loss*. 2024. URL: https://www.hopkinsmedicine.org/health/wellness-and-prevention/the-hidden-risks-of-hearing-loss#:~:text=In%20a%20study%20that%20tracked,more%20likely%20to%20develop%20dementia..

[21] Ltd Shenzhen Normand Electronic Co. *Specification*. 2021. URL: https://cdn-shop.adafruit.com/product-files/3094/Datasheet.pdf.

[22] Freda Su. *Date Calc*. 2022. URL: https://courses.ideate.cmu.edu/60-223/f2022/work/date-calc/.

[23] Watches of Switzerland. *A Watch Size Guide*. 2024. URL: https://www.watchesofswitzerland.com/watch-buying-guide/watch-size-guide.

[24] Espressif Systems. *ADC*. 2024. URL: https://docs.espressif.com/projects/arduino-esp32/en/latest/api/adc.html.

[25] Rob Tillaart. *RunningAverage*. 2024. URL: https://www.arduino.cc/reference/en/libraries/runningaverage/.

[26] Jayme-Lee Tolliday. *What are A, C, Z Frequency Weightings?* 2020. URL: https://www.cirrusresearch.co.uk/blog/2020/03/what-are-a-c-z-frequency-weightings/.

[27] Eugene Ungar and Kenneth Stroud. *A New Approach to Defining Human Touch Temperature Standards*. 2010. URL: https://ntrs.nasa.gov/citations/20100020960.

[28] Wikipedia. *Decibel*. 2024. URL: https://en.wikipedia.org/wiki/Decibel.

[29] Wikipedia. *Equal-loudness contour*. 2024. URL: https://en.wikipedia.org/wiki/Equal-loudness_contour.

[30] *WolframAlpha*. 2024. URL: https://www.wolframalpha.com/.

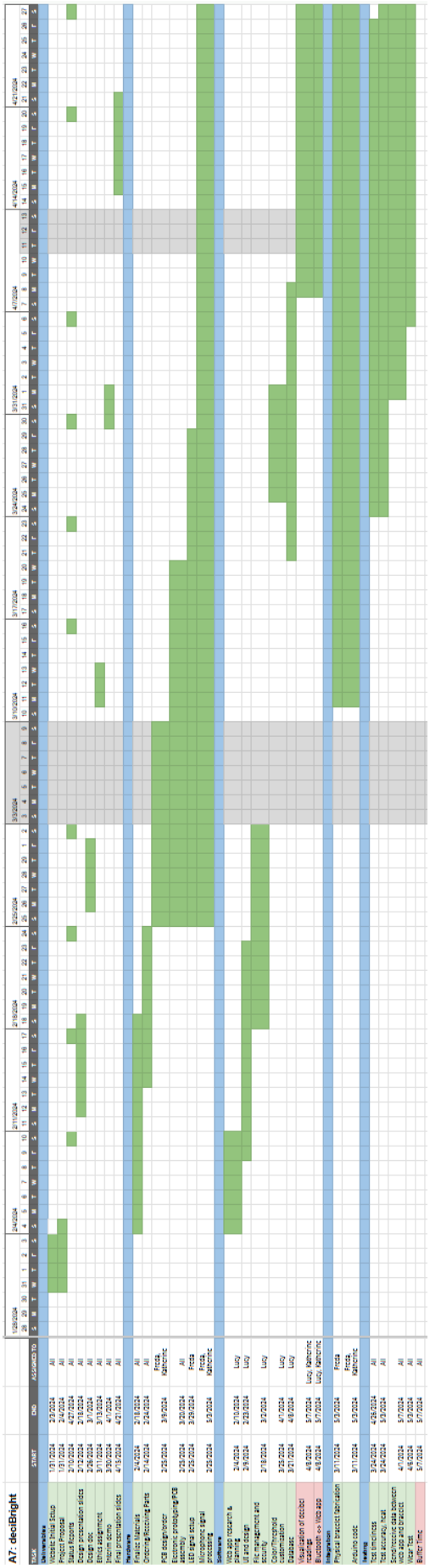

Figure 39: Gantt Chart