Twinkle Tines

Final Design Document (Spring 2023)

Team Number: sdmay23-30

Client & Advisor: Dr. Mani Mina

Team Members:

Mesa Hassel (SE) Eileen Hillier (EE)

Kaitlyn Nolting (SE) Stuart Pearson (EE)

Andrew Adams (SE) Daniel Duerr (SE)

Isaac Vrba (CPRE E)

Team Email: sdmay23-30@iastate.edu

Team Website: https://sdmay23-30.sd.ece.iastate.edu

Revised: 5/1/2023

Executive Summary

Preamble

A kalimba is a musical instrument (example shown in figure 0.1), usually a wood base with metal tines, that is played by plucking the tines, making a sweet bell-like sound. Due to the thumb dexterity required to play tines that are so close together, it is almost impossible to play without looking directly where your thumbs are. This makes playing the kalimba while reading sheet music exceedingly difficult, as one can't simultaneously focus their eyes in two places multiple inches apart at once; thus, memorization is the typical route to learning songs. Memorizing songs is more effort than many casual users would like to put in, so an idea for a senior design project was pitched whereby some light indicators on the tines themselves would illuminate the notes in the correct order of a song so that a user could successfully play songs without having to memorize music.

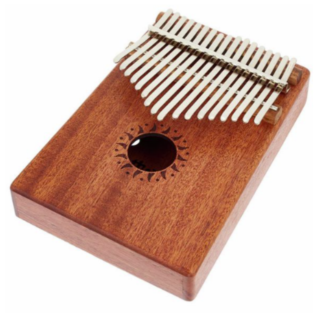


Figure 0.1

Development Standards & Practices Used

This project originated with a student submission. In the beginning, the design was only an idea that has evolved. The final design will continue to shift as further testing is completed. With that in mind, hardware and software practices alongside engineering standards have changed as the project has progressed into its initial implementation.

Hardware Practices and Standards:

- 6.35 mm audio jacks
- 3.5 mm audio jacks
- microcontroller GPIO Communication

Software Practices:

- To properly test the software we will adhere to the IEEE 829 Software Testing Practices
- Following standard Git techniques and file management.

Engineering Standards: The following standards are currently under consideration

- FCC PART 15.247, Federal bluetooth design standards for low power 2.4 GHz bluetooth communication.
- IEEE Standard for Bluetooth Low Energy (BLE) devices 802.15.1
- IEEE Standard for Documentation Schema for Repair and Assembly of Electronic Devices

Summary of Requirements

- Interface a kalimba with software for song note instruction
- The software should store accessible songs in a library/database
- The lighting system should illuminate notes (tines) to be played in a correct note order
- The interfaced hardware and software should correctly recognize tines played

- The mount system should be transferable

Applicable Courses from Iowa State University Curriculum

Hardware Application Classes:

- CPRE 288 - Embedded Systems I
- CPRE 488 - Embedded Systems Design
- EE 224 - Signals and Systems I
- EE 201 - Electric Circuits
- EE 230 - Electronic Circuits and Systems
- EE 535 - Physics of Semiconductors
- MATH 267 - Differential Equations (for FFT)

Software Application Classes:

- ComS/SE 309
- ComS 363
- SE 319

New Skills/Knowledge acquired that was not taught in courses

Here lists all our new skills/knowledge that our team acquired which was not part of Iowa State's curriculum that was needed in order to complete this project.

On the hardware side 3d modeling/3d printing and soldering skills were developed by team members that did not previously have that skill, mostly because these skills are not taught in the regular ECPE curriculum.

On the software side we had to research a couple different things. This included bluetooth connection to a microcontroller, react-native, firebase database. Everything we plan to do on the hardware side is at least tangentially related to things that we have done in class thus far. It is mostly design soft skills that we have learned so far. Both the hardware and software team had to develop soft skills such as:

- How to make a long term effective project plan
- How to effectively manage work planning with interfering schedules
- How to manage a client/advisor relationship
- How to create a bill of materials and order equipment.
- Strategies for effective documentation and testing.

Table of Contents

List of figures/tables/symbols/definitions

1 - Team

1.1 Team Members

- Mesa Hassel (SE)
- Andrew Adams (SE)
- Stuart Pearson (EE)
- Eileen Hillier (EE)
- Kaitlyn Nolting (SE)
- Daniel Duerr (SE)
- Isaac Vrba (CPR E)

1.2 Required Skill Sets for Your Project

Knowledge about microcontrollers and how they interact with inputs and outputs

Basic circuit and amplifier theory

Basic circuit construction and testing techniques

Basic signal processing theory

Soldering skills

3d modeling/ 3d printing

Basic understanding of Music Theory

How to interact with backend databases such as using a song library

Building basic app user interfaces

Understanding of Bluetooth Low Energy Systems

A good attitude and sense of fun

1.3 Skill Sets covered by the Team

Knowledge about microcontrollers and how they interact with inputs and outputs

- Isaac, Stuart, Eileen

Basic circuit and amplifier theory

- Isaac, Stuart, Eileen

Basic circuit construction and testing techniques

- Isaac, Stuart, Eileen

Basic signal processing theory

- Stuart, Eileen

Soldering skills

- Isaac, Kaitlym, Stuart

3d modeling/ 3d printing

- Eileen, Stuart

Basic understanding of Music Theory

- Andrew, Kaitlyn, Stuart, Daniel

How to interact with backend databases such as using a song library

- Andrew, Kaitlyn, Mesa, Daniel

Building basic app user interfaces

- Andrew, Kaitlyn, Mesa, Daniel

Understanding of Bluetooth Low Energy Systems

- Kaitlyn, Isaac

A good attitude and sense of fun

- Stuart :) , Isaac (: , Kaitlyn :) , MESA, Daniel, Eileen

1.4 Project Management Style Adopted by the team

Our team participated in an Agile framework of project management. We had large milestones that took longer than one sprint but our weekly meetings naturally fit with single week sprints. Each meeting we demoed our current progress and developed new ideas for the next week/sprint. Since our final destination was flexible and initially somewhat nebulous, multiple design process iterations of the agile framework allowed us to continuously refine our ideas and final design.

1.5 Initial Project Management Roles

- Isaac Vrba — *Hardware Programming and Device Setup*
- Mesa Hassel — *Firebase Database and UI Implementation*
- Stuart Pearson — *Note Detection Mechanism*
- Eileen Hillier — *3D Printed Mount*
- Andrew Adams — *Database Querying*
- Kaitlyn Nolting — *React-Native Bluetooth Implementation*
- Daniel Duerr — *Documentation*

2 - Introduction

2.1 Problem Statement

How might we display the correct order and timing of notes on the instrument for new kalimba users so they don't have to look back and forth between the instrument and the music sheet to play a song?

Who has the problem?

- New kalimba users, kalimba users learning a new song.

What is the problem?

- It is hard to learn songs while looking back and forth at the music sheet and the instrument, and memorizing can be laborious.

Where is the problem occurring?

- In the inability to easily focus on two different places at once while mastering a new skill.

When is the problem occurring?

- When a kalimba user is trying to learn a new song.

Why is it important?

- It will save time, and make learning new songs easier/less laborious.

How will it be solved?

- Adding light features that show which note is to be played next and when they are to be played.

2.2 Intended Users and Uses

Persona (1): Someone with music experience that wants to enjoy picking up a Kalimba and playing music without putting in a lot of time or effort. Someone looking to reduce stress.

Needs (2) and User Benefits (3): A person, who has an interest in music, needs an easier and faster way to learn how to play the kalimba because they have a hobby for learning new skills, and they are struggling to learn efficiently without giving effort beyond what is desired.

This is the intended user for our project at this point, should we have time in the future to address the needs of other personas (i.e. maybe four-year-olds or advanced musicians) we can. It will be easier to design for the general user first and then tweak features later to address non general users than it would be to design for a bunch of different users at once. Obviously, the general user (i.e. us) that would benefit from Twinkle Tines and anyone who enjoys the music played would receive the secondary benefits. They will also have reduced stress and frustration by easily being able to play music and relax. The creators would also benefit intrinsically from seeing people enjoy the fruits of their labor.

- **hobbies & interests:** the user enjoys music/instruments
- **demographic**: open to anyone with new features added, needs a working finger
- **motivation:** supposed to give anyone instant gratification for being able to pick it up and play a song

- **personality & emotions:** creative personality, hopeful, adventurous, like to learn new things
- **values:** gain new skills and enjoyable distractions/relaxations

Empathy Map

Who are we empathizing with?
- new kalimba users

What do they need to do?
- learn to play songs on the kalimba
- play songs soundly
- be satisfied with their own ability

What do they see?
- reading music notes, sheet music, and youtube videos of others playing the kalimba

What do they say?
- "I want to be able to play a song?"
- "I want to learn a new instrument?"
- "That looks fun to play!"

What do they do?
- attempting to learn songs, researching how to play, struggling to play songs while also reading the sheet music

What do they hear?
- others playing on youtube, playing runs, the choppiness of trying to play songs themselves

What do they think?
- it's difficult, it's harder than I thought

What do they feel?
- frustrated
- confused

2.3 Requirements & Constraints

Functional
- The next note to be played should light up no longer than a quarter of a second after the correct note is played.
- The lights should be bright enough to be seen in general lighting conditions.
- The hardware needs to be able to recognize that the correct note/group of notes has been played within a quarter of a second.
- The software should display the next note(s) to be played.
- The software should be able to pull songs from a set of stored songs within the database.
- The lights shall not block or limit the user's ability to play the Kalimba.

Resource
- Any mobile device capable of running android or ios shall be able to run the app.
- There should be an external LED mount device to light up and detect notes played.
- The application shall connect to the hardware device via Bluetooth Low Energy

Physical

- The hardware apparatus should be no heavier than half a pound
- The hardware apparatus should not increase the overall footprint by more than 10 percent of the original kalimba size.
- The LED apparatus shall not interfere with the user's ability to pluck the tines

User experiential

- The user should be able to access the library of songs
- The user should be able to favorite songs from the extensive library page
- The user should be able to connect their application to the hardware device
- The user should be able to start the song by clicking a button

3 - Project Plan

3.1 Project Management/Tracking Procedures

The Agile scheme has allowed us to restructure and change goals or functionality as we learned more about how our specific implementation was going to come together. We planned this to be a sprint heavy project to get rapid feedback and development, and the agile style will served that strategy best.

For the software side we will collaborate using github to share our code progress with one another. We also utilize the user story framework within github to be able to break down our progress into smaller chunks and better implement our agile scheme. Hardware and overall team progress will be tracked by hand on a shared google document/drive. Team communication also happens through Discord.

3.2 Task Decomposition

1. Hardware
    a. Research Period
        i. What kind of LEDs will be best for the design?
        ii. What method will we use to control the LEDs
        iii. What method will we use to interface the LEDs to the control system
        iv. What method will we use to detect the notes being played
        v. mounting apparatus form
    b. Vibration/Pitch Detection
        i. Correct detection of note played
        ii. Detecting correctly multiple notes played simultaneously
        iii. Location of device on kalimba
    c. LED Testing
        i. Variable lighting control test
        ii. Brightness control test
        iii. Location of LEDs
    d. Communication with Software
        i. When to light next LED
        ii. Correction passing of vibrational information from hardware to software

2. Software
    a. Research Period
        i. How to store/upload music
        ii. How to control the LEDs through software
    b. Design UI Period
        i. Figma app layout
        ii. create initial app pages
    c. Core Functionality
        i. add songs to a database
        ii. create a function for users to add/remove songs to a local library
        iii. create a function for users to preview songs

        iv. light the LEDs from the software according to the next note
    d. Test Software
        i. create tests to check for any bugs that may need to be fixed

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Project Milestones

- Hardware parts are ordered before the start of Thanksgiving Break
- Successful hardware unit test of detecting input and lighting up the corresponding LED to demonstrate the working relation between our audio detection and LED.
- Hardware and Software Complete Integration
- Complete Working Prototype
- Complete Project with our final design and filled out documentation.

Hardware

- Designers should be able to select methods and devices for LEDs, LED control, LED interface, note detection, and mounting apparatus upon conclusion of the research period.
- LEDs are able to light up and indicate notes to be played when selected within 0.25s
- Note detection device should be able to detect notes with 95% accuracy and 0.15s
- Device should communicate information to and from software within 0.1s
- Device LED apparatus should be mountable on most normally sized and shaped kalimba

Software

- Note recognition software should be able to identify 95% of the notes played by the user. (shared milestone)
- Software should be able to assign each note to its corresponding variable with 95% accuracy.
- Users should be able to add songs to a local library.
- Be able to access all pages in the app from any other page.
- Software should be able to light the LEDs for each note.

3.5 Risks And Risk Management/Mitigation

General Risks:

- [**0.001**] React-Native is no longer supported
- [**0.002**] Bluetooth is no longer available

Research:

- [**0.05**] Research incorrect information and need to restart or come back later and do more research.
- [**0.1**] We are unable to come to a conclusion on hardware due to not being able to find a solid solution

Purchase Hardware Devices:

- **[0.3]** Purchase incorrect or poor quality device, risk should be mitigated through research phase and purchasing extra devices
- **[0.1]** We already have two kalimbas, and generally LEDs are very cheap and available, low supply chain risk
- **[0.25]** If we need a PCB, from a quick google search, lead time is usually not more than a month and often much less for simple boards. Additionally, the PCB could be constructed wrong and have to be resubmitted for manufacturing.

Figma App Design:

- **[0.2]** Can't access Figma.
- **[0.4]** Figma has significant learning curve

LED Interfacing Testing (Phase 1 Breadboard setup):

- **[0.2]** One or more of the LEDs could be bad apples, so we will get many extras to prevent this from being an issue

Create User Interface:

- **[0.1]** Designs from the app design phase do not work in the program.

LED Interfacing Testing (Phase 2 Final Part Setup):

- **[0.15]** LEDs overheat and burn out, risk should be mitigated in phase 1.

Create Database of Songs:

- **[0.1]** Can't store songs locally

User uploading songs functionality:

- **[0.2]** No file format is compatible with the method of uploading we chose

Note Detection:

- **[0.05]** On a kalimba that is improperly tuned, the device might correctly identify a "wrong note." The solution would be to retune
- **[0.45]** Too much background noise. This should theoretically be completely mitigated with a pickup
- **[0.2]** Complications while attempting to record the note that the user played.
- **[0.35]** Multiple notes at once may not all be registered if one of the notes played is especially quiet.

Hardware & Software connection:

- **[0.3]** Incompatibility between connections, risk mitigated through early communication between teams
- **[0.2]**Either the hardware or software team is not ready in time for our planned connection between the two in March.

Playback functionality:

- **[0.25]** Bugs in the code may crash/freeze/loop software in the middle of playback.
    - Hard to judge probability due to the heavy dependence on how we code it

Final Testing:

- **[0.99]** Someone gets the urge to violently hurl the kalima at the floor
    - Risk Mitigation: Good thing we have a second one!
- **[0.05]** Catastrophic failure at the very end, low risk
- **[0.05]** At any point we could drop the Kalimba near the end, break everything, and not have time to rebuild. This can be mitigated by developing multiple kalimbas and keeping one as a dedicated back up.

3.6 Personnel Effort Requirements

Andrew Adams	**Isaac Vrba**	**Eileen Hillier**	**Kaitlyn Nolting**	**Mesa Hassel**	**Daniel Duerr**	**Stuart Pearson**
4 hours/ week	4 hours/ week	4 hours/ week	4 hours/ week	4 hours/ week	4 hours/ week	4 hours/ week

Figure 3.2

Individual effort requirements will be subject to change based on subgroup and timeline date. The group will also discuss the need to increase the number of hours when facing increasing project demands. Hours include 1 hours per weekly meeting.

3.7 Other Resource Requirements

- LEDs, Kalimba, communication cords, 3d printing access for mount, electrical interfacing materials
- Our undying love for one another
- Potentially advice from people with music expertise
- Research sources
- Copious Caffeine Collections

4 - Design

4.1 Design Context

4.1.1 Broader Context

Area	Description	Examples
Public health, safety,and welfare	How does your project affect the general well-being of various stakeholder groups? These groups may be direct users or may be indirectly affected (e.g., solution is implemented in their communities)	- This project may reduce stress for users - This project may provide a creative outlet for some - This project could provide a relatively healthy coping mechanism for some. - This project could increase general welfare for the population from the increased enjoyment from easily learning the kalimba
Global, cultural, and social	How well does your project reflect the values, practices, and aims of the cultural groups it affects? Groups may include but are not limited to specific communities, nations, professions, workplaces, and ethnic cultures.	- This project could potentially increase musical interaction for subgroups of people that would not otherwise want to put in the effort required to develop musical skill. - More children might start to interact music as they could easily learn to play familiar songs - More people might start replacing short bursts of screen time with Kalimba time. - This project might bring more recognition to Zimbabwe in Africa, where the kalimba is thought to have originated
Environmental	What environmental impact might your project have? This can include indirect effects, such as deforestation or unsustainable practices related to materials manufacture or procurement.	- Increase to the creation of non-renewable materials through the use of plastics/metals required for the project - Slight increase to indoor noise pollution by encouraging people to play the kalimba - Marginal increase to deforestation for material to create kalimbas

Economic	What economic impact might your project have? This can include the financial viability of your product within your team or company, cost to consumers, or broader economic effects on communities, markets, nations, and other groups.	- The product needs to be affordable for any user as the Kalimba itself is relatively cheap and the device we create shouldn't cost much more than the instrument itself. - This project could potentially open up a previously untapped market.

Figure 4.1

4.1.2 Prior Work/Solutions

Some similar products exist for different instruments such as the piano. Although these products exist for other instruments there would not be any competition since there have not been any devices made for the Kalimba as far as we know. Some cons could be that our device would require another attachment whereas the piano has the light-up function as a built-in feature. This could also be seen as a pro as well since the user could use the device with multiple different kalimbas. The user could also easily have one kalimba with the light-up device and one without.

For hardware, there exists light-up pianos that operate similarly to our light-up kalimba.

- Here is one example of this.
- and here is an example video of one in action coupled with software.

For software, there does not exist an exact copy of what we are going to be using, but there is a very good visual tool for learning online that we found called TabWhale. Instead of being web-based, our product will include a mobile application with the tutorial song library. Another source of inspiration on the software side is KalimbaTabs.

4.1.3 Technical Complexity

1. The design consists of multiple components/subsystems that each utilize distinct scientific, mathematical, or engineering principles –AND–

   There is a large hardware and software component that each contain many subsystems and all connect to one another. For software, there will be the frontend subcomponent that will contain all of our UI design, and be able to pull information from the backend. There will also be a backend subcomponent that will store our various songs in a database for the frontend to use.

   For the hardware side of things, we are going to be dealing with components such asa pickup to transfer vibrations to the software for note reading, LEDs to visually display to the user the next note(s) to play. There would also be the wiring for both of those, and we are going to try and make it so at the end, our device will connect to a phone through a single usb type c cable or via bluetooth

2. The problem scope contains multiple challenging requirements that match or exceed current solutions or industry standards.

At this current time, there does not exist a tool that will allow as much direct interactivity. Our tool will mount any kalimba and through a series of hardware mounts and software I/O, the user will be able to get instant feedback from the device to more rapidly contribute towards their learning.

The closest way someone is able to mimic what we are trying to do is to write down tabs and place those written notes near the top of the kalimba while attempting to play the tines in order. There is no tool currently available that gives the user immediate feedback while they play notes.

4.2 Design Exploration

4.2.1 Design Decisions

- The device that we create should be physically transferable between different kalimbas, so that someone can buy one device and attach it to any typical kalimba size.
- The device will also use a pickup to determine which notes the user has played. This sensor type was chosen to reduce the chance of outside interference that could occur while only using an audio sensor for note detection. Having both sensor types increases the accuracy of our detection system.
- The software will use some existing music library, for example musicXML, that will make it possible to read the notes from the sheet music in the app, we have not yet decided on what we will use there is still research we need to do to figure out which is best for our project

4.2.2 Ideation

In order to decide between the following options, we brainstormed ideas together as a group. As mentioned in the following section, the pros/cons that we came up with ruled out many of our options.

- Design Decision: Recognizing that a specific note is played
- Five Potential Options:
  1. **Vibration**. We could directly pass the vibrational information from the hardware to the software. This could be done with a simple pickup.
  2. **Sound**. The software would be able to recognize the tune of the tine that was played via a mic and determine whether it was the correct note or not based on tunes set up in the database.
  3. **Tine Being Touched**. The hardware device would detect any time the user touches a tine to indicate a note being played. (i.e. a sensor physically attached to a tine)
  4. **Tine Movement/Proximity**. For this idea there would have been physical sensors next to each tine that would detect when a specific tine is plucked. (i.e. a sensor physically very close to a tine)
  5. **Optically Detected Movement.** A camera would be set up to visually detect when the user has plucked each tine.

4.2.3 Decision-Making and Trade-Off

In order to identify our pros and cons for each of the options we brainstormed ideas at our meeting. The cons ruled out most of the options and we were easily able to pick the best option.

Vibration

- Pros: Sensors already exist for the Kalimba to transmit this, and we already have a pickup to test with. It would also be easy to transmit this information with an aux cord, and the power draw would be very small.
- Cons: The pickup might not have the sensitivity to correctly identify notes played, especially if many are played at once. If the user is very percussive in their playing, they might introduce noise that messes with the program a lot.

Sound

- Pros: This would be fairly simple to program. Most modern devices have mics prebuilt in them that could be used.
- Cons: If a key is out of tune and isn't in the database the software may not be able to recognize it as the correct note and the user would be stuck on that note. Also, any background noise in the user environment could mess with the note recognition, which is a big problem..

Tine Being Touched

- Pros: Straightforward to detect when any tine is played.
- Cons: Overly sensitive to user input, would often mistake the user resting their fingers with a note being intentionally played. This would also potentially be more expensive as many sensors wouldn't need to be used and powered. Could impact the vibration of the tines which would ruin the sound of the instrument.

Tine Movement/Proximity

- Pros: Straightforward to detect when any tine is played.
- Cons: It could be difficult to differentiate between the plucking movement and other movements the user makes that is not meant to actually pluck the tines. It would be hard to tell when the tine stops producing sound depending on the location of the sensor. This would also potentially be more expensive as many sensors might need to be used and powered.

Optically Detected Movement

- Pros: Because it is using visual input, it would be able to take in lots of information about the tines being plucked.
- Cons: This would require setting up an external camera, and the external camera would also have to connect to the rest of the system, it would also be hard to see which tine the user is touching since they are close together and it may look like the user is touching more than one, it could also think the user has played a tine if

they are touching it even if they haven't actually played it. This would also be expensive in terms of power draw and equipment.

4.3 Proposed Design

4.3.1 Overview

High Level Functional Block Diagram

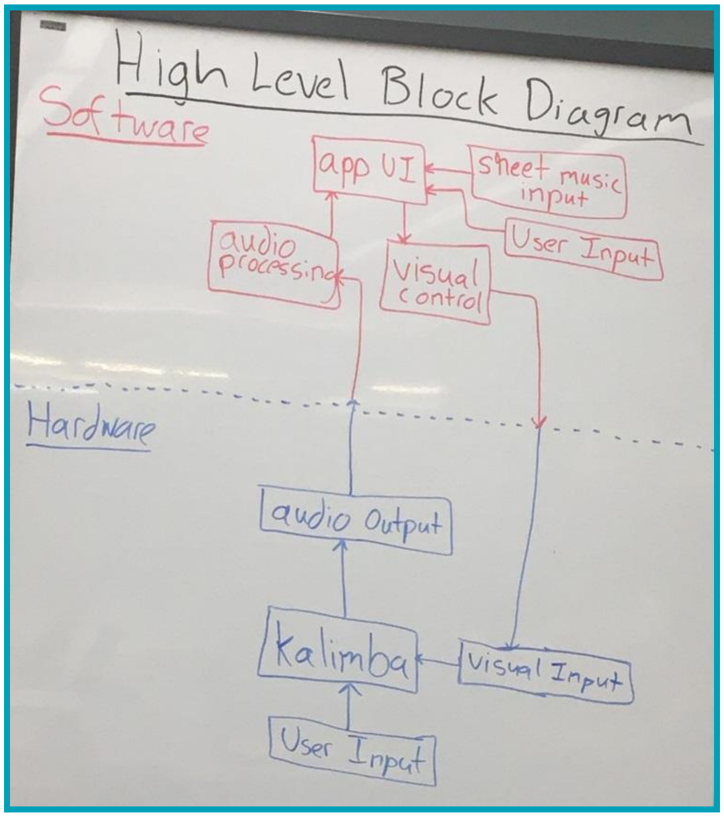


Figure 4.2

The software side of our design will be a mobile application where a user can add/remove songs to their own liked library. They can add or remove songs from the app's existing library or upload sheet music that will then be processed to be able to be played. They will then be able to play songs from this library, the notes of this song will then be displayed on the device that is connected to the kalimba. There will be a led light over every tab on the kalimba that will light up based on which note the user needs to play next. This device will also be able to process which note the user has just played and whether it was correct or not. Once this is determined the software will decide if the next note in the song will be played or if the same note will display again.

4.3.2 Detailed Design and Visual(s)

Engineering Specific Block Diagram

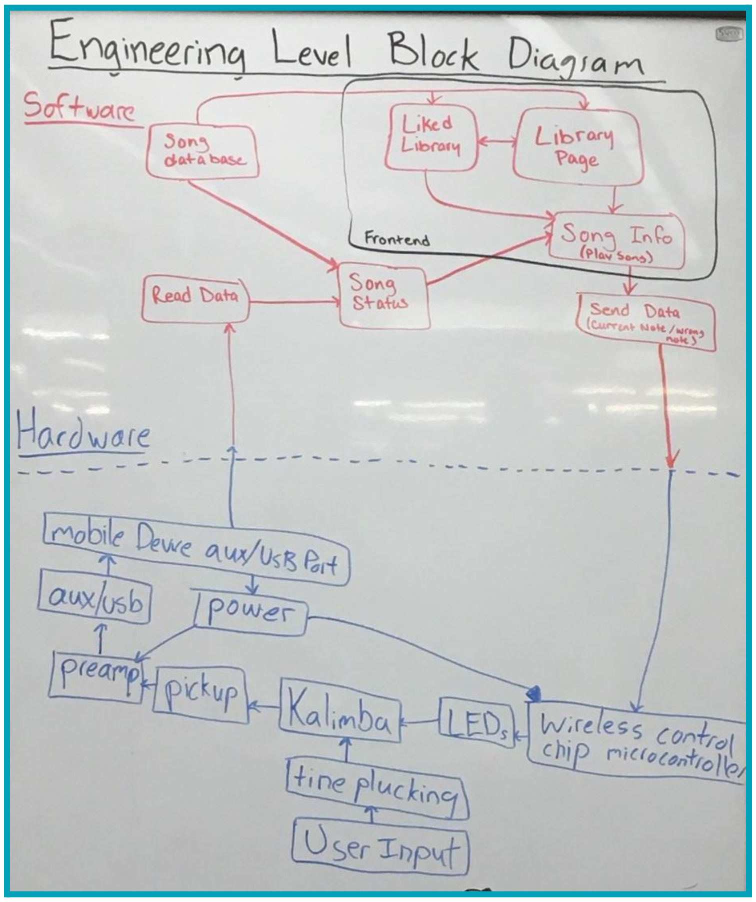


Figure 4.3

The hardware side of the device would consist of inputs of an array of LEDs controlled wirelessly with a microcontroller chip and user inputs from the plucking of the tines on the kalimba. The audio output would be gathered with an audio pickup fed into a preamp in order to make the signal intelligible for software to read it. This will then be fed into an aux/usb port which will connect to the user device (generally a smartphone).

For the backend side of the software, the mobile application has a database that stores songs. A user can select which song to play from the library, and this will query the backend to choose that specific song based on the song ID. The backend will send the song name and the individual song notes stored as a string array. Each note is represented by a specific number. These numbers tell the physical device which specific note to turn on. Once the note that the user played is determined the backend will take in that information and compare it with the note we are on in the array and determine whether the next note should be lit or if we stay on the current note. (A tine will stay lit until the user correctly plucks that tine)

For the frontend side of the software, the mobile application will have a page that will have a user's library for songs they have added to it as well as songs that we added, a page/function where they can upload sheet music and a way to start a song on the physical device.

4.3.3 Functionality

Step 1: The user mounts the device to the kalimba.

Step 2: The user applies a pick up if one is not already installed

Step 3: The user connects the pick up to the preamp if a preamp is used.

Step 4: The user connects the preamp to the appropriate port on the mobile device if a preamp is used.

Step 5: The user connects power to the microcontroller via the appropriate port on the mobile device

Step 6: The user opens the app on the phone.

> **Step 6.1:** The user can add songs to their liked library from the larger library
>
> **Step 6.2:** The user can view the details of a song

Step 7: The user selects a song to play on the application.

Step 8: The application sends the note data to the kalimba to light up the tines

Step 9: The user plucks the lit up tines in the order they are presented based on the note order of the preselected song.

Step 10: Once the user has finished a song, they can select another song to play or disconnect the power to the device and unplug the preamp.

Storyboard of the User - Our project with a Kalimba by Isaac Vrba

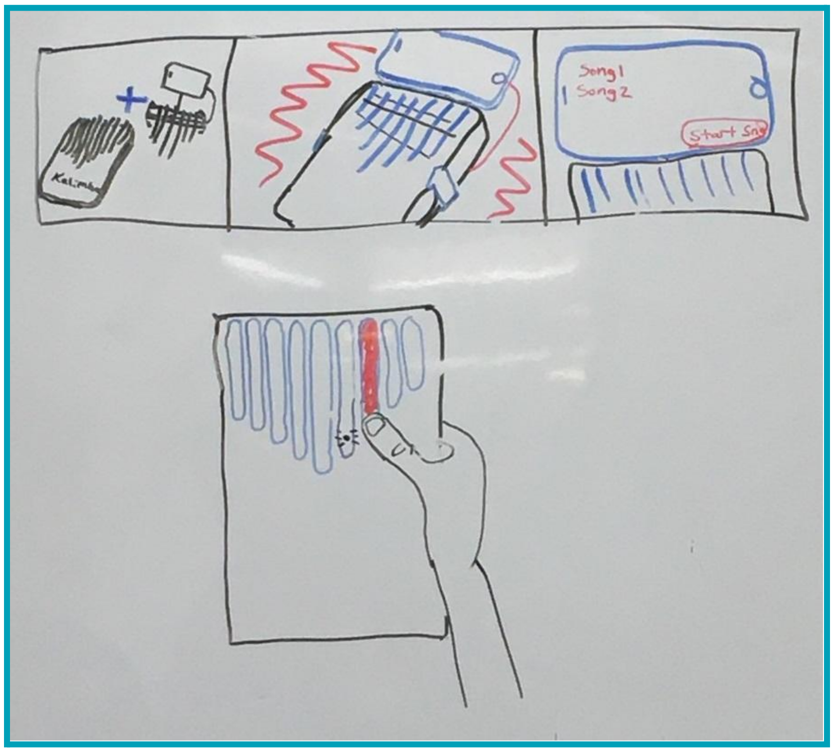


Figure 4.4 - Tickling the Tines - Isaac Vrba

4.3.4 Areas of Concern and Development

We have tried to keep user needs a top priority when creating our current design. The main requirement our user needs is to be able to easily learn how to play songs on the kalimba. Our apparatus does this very well at a high level by easily allowing users to be able to choose certain songs, and then correctly lighting up the notes with the LEDs so that the user gets feedback when they play correct or incorrect notes.

Our primary concerns for our current design include the method of attachment, ease of user set up, and the ability for users to upload their own songs. We have not tested a method of attachment of the device to the kalimba and have not settled on a concrete idea to begin testing. The device and user set up should be designed to maximize ease of use, our current set up has not been streamlined yet for this capacity. We would like for the user to be able to upload their own songs to our app, this could present difficulties with reading the music and correctly formatting the song to play on the kalimba.

As far as the mounting attachment goes, we need to wait until we decide on what hardware that we are going to use (LEDS, amps, and microcontrollers). We are currently still working on how the design will come together, and we specifically have set up meetings with professors that are experts with analog electronics and microcontrollers to gain some insight on how to best meet the needs of the users with the hardware that we want to implement. These questions in the meetings will mostly be centered around microcontroller and amplifier selection. When this is all done, we will be able to work on the mount.

4.4 Technology Considerations

Hardware technology

- LED
  - LEDs allow us to locate a lot of lights in a small space, perfect for the small design of the mount used for the Kalimba. We wanted to make sure the user is able to see the upcoming notes and any mistakes if they play the wrong note. We think that this is the best visual we can provide to the users.
- Pre-amp
  - A strength of the pre-amp is that it makes the sound waves much more readable for a pickup or condenser microphone to pick out each individual note. A weakness, however, would be that a pre-amp is another separate piece of hardware that we would have to add to our design which would increase the cost as well as the size of our entire contraption.
- Wireless microcontroller control chip
  - We are still not completely sure of what model we are going to use, however the particular model we are going to use will be discussed and purchased by the end of our research period. The strengths of the wireless microcontroller include the robustness of the device, allowing us to plug in multiple hardware components and communicate to everything in an easy manner.
- Pick-Up
  - What is a Piezo pickup and why do we need an amp? Piezoelectric amps work due to the piezoelectric effect. Piezoelectric materials, usually a type of crystal or ceramic, produce a voltage in response to mechanical compression. So a piezoelectric material can produce a voltage when squeezed and have the voltage disappear when the squeezing ceases. Just as sound waves are pressure waves in the air, vibrational waves are pressure waves in a material. So, when a musical instrument is making a sound, it is vibrating. By attaching a piezoelectric pickup to an instrument, the piezoelectric material in the pickup will vibrate at the same frequency as the instrument, and thus, the piezoelectric material expands and contracts at that vibrational frequency and a voltage will appear that perfectly matches the sound frequency. However, the output impedance of the piezoelectric material is usually very high (the piezoelectric material is not a conductor after all), which means the potential output voltage that a computer would have to read would be very small to the point a computer might not even recognize that a voltage signal is present. So the signal needs an appropriate amplifier between the piezoelectric material and the computer for the majority voltage drop to occur across the load (i.e. computer) instead of the piezoelectric material. (Ask me what a voltage divider is to properly explain impedance matching.) Anecdotally, when I plug the kalimba into my roommate's guitar amp and turn the volume all the way up, it is still really quite relative to connecting a guitar. This happens because the

output impedance of the piezoelectric pickup is still large relative to the load impedance of the guitar amp.
- Explanation video

Software technology

- Song database
    - there will be a lot of songs that the user can choose from and data for each song so this will be easiest to work with if stored in a database
- External music sheet files
    - musicXML, midi, etc.
        - There are multiple strengths and weaknesses for each external music sheet file type we have considered.
            - Midi would be a lot easier to read from, because the notes can easily be changed into an array of data. However, we would need to create midi for each song or find a library of midi arrangements for each song.
            - Sheet music would have the strength of it's very easy for a user to find, and they could easily upload it. However, a weakness could be that sheet music may be more difficult to parse when changing the notes to an array of data.
    - The purpose of only accepting specific file formats is so that our software can easily read it
- React native
    - React native does well to support designing an application that supports both ios and android
    - React native has lots of supporting documentation and useful resources to be able to learn how to use it for our system.
- Git
    - We decided that git would be a good technology to use for version control because it does a very good job of collaborating on teams with code, and we all have a lot of experience using it.
- Tune function
    - this function will pick up the tune when a user plays a note
    - this can be used to detect whether a user played the right or wrong note instead of using the pre-amp
    - this function could also be an in app function the user can use if their kalimba is out of tune which is important for our app to work overall

4.5 Design Analysis

A pickup for the kalimba has been tested. A piezoelectric pickup was attached to the kalimba and then subsequently plugged into a guitar amplifier. The guitar amplifier has to be turned up to full volume for a soft output sound to be heard; although, the output sound was just as clear and full as the kalimba itself. After some research, it has been determined that a preamplifier is going to have to be added to our design in order for the software to receive a waveform that can actually be used. The issue is that the output impedance of the piezo pickup is high, around 1 Mohm, and the input impedance of most amplifiers is not relatively high, which is why the output on the guitar amp was still soft despite being at max volume.

We have yet to test other aspects of our design, but subsequent testing may alter future plans just as testing with the piezoelectric pickup and the guitar amp has shown that we will need a preamp for the pickup to be effective.

5 - Testing

5.1 Unit Testing

<u>Hardware</u>

LEDs - The LED arrays selected for the project have been vetted to work as expected.

Microcontroller - The microcontroller and onboard PDM microphone have been tested and vetted to work as they are supposed to

Sensors - Multiple sensor devices were tested including pick up, onboard microphone, and onboard accelerometer to detect and measure the audio/vibrational output from the kalimba. All devices were tested and vetted to work on their own in isolation

<u>Software</u>

Song Database - test that queries to the database grab the correct data (song and song info). We used a mock database to be able to use with unit tests created in Jest.

UI - test that each page is reachable from the homescreen. We created several unit tests in the code to test the functionality and that the proper elements were showing up. We also tested this manually a lot.

Song Favorites - test that the user can favorite a song to add it to their favorites list. We used a mock database to ensure that the querying of data was working correctly.

Library - test that the user can view all songs in the library and test for the specific components within the library. We used a mock database to ensure that the querying of data was working correctly in the library as well.

5.2 Interface Testing

User Interface

- Test that all of the screens in the app are reachable from the homescreen/landing screen.
- We will also need to test that each button or interactable feature produces the intended result.
- Verify that our UI is rendering according to our initial screen sketches produced in Figma (or other UI software).

Physical Interface

- The response time has been tested and vetted to be within the goal .25 second.
- The mount has been implemented and vetted to work as needed. There are optimizations that can be made though

5.3 Integration Testing

Software Hardware Integration

- **Note Detection** - This is tested by using a pickup to physically wire vibration frequency information to a preamp and then on to the software and via an onboard microphone on the microcontroller to send the vibrational information to the software. Tools included in this testing will be LEDs, pickup/microphone, breadboard, power supplies, oscilloscopes, the kalimba itself, microcontroller, software mobile service, and the user themself.
- **BLE Connection** - tested the connection by running the app and trying to connect with the Hardware. We did a lot of manual testing with the Hardware team to make sure the connection was working and data was being sent since the type of data and setup was pretty specific to the setup of the Hardware bluetooth code.

5.4 System Testing

End-to-end testing should ensure complete and fast communication between hardware and software systems. System testing would require software-hardware integration, user interface button testing, physical interface input testing, and song database unit testing. System is required to light up LEDs in correspondence with song database order and user input timing while also following commands from the user interface buttons. This would be tested through the combined usage of tools needed in previous testing phases and rigorous user testing.

System testing was done with dummy data to vet that the necessary communication occurred as designed.

5.5 Regression Testing

Once we are at the point when we are ready to tie individual units together, we can begin regression testing. We will not have many different units tying together (audio system, lighting system, communication system, and software), but everytime that a new unit is integrated, we will have to test whether everything else still works. For example, on the hardware side, once we can get to the point where the microcontroller is able to recognize specific notes that are played, we will next add the LEDs as visual feedback that specific notes were played. Once the LEDs are added, we will have to ensure that the system is still recognizing the notes correctly.

Regression tests were completed as necessary as the project continued.

Using Git we will be able to have many versions of our software available in case a new component broke the previous functionality. We can also implement a CI/CD pipeline to test new software as it is added to our Git repo.

5.6 Acceptance Testing

We will demonstrate our design requirements by sticking to the fundamentals that we want to enable a person to use our device on a kalimba and be able to play a song from start to finish, and with enough practice could learn how to play the instrument. We have then been taking decisions that will help us meet that goal by using visuals via LEDs and music notes detection for accurate note recognition to allow us to give the user timely and constructive feedback.

For any decision that strays off of our original plan is to be discussed within each team, hardware or software, to discuss if changing plans or testing different components will result in a better user experience while still following our functional and non-functional requirements. We will involve our client for acceptance testing while we are developing our prototypes and ask them for feedback during major milestones.

The final demo video serves as the acceptance test showing that the design works as intended and is ready for presentation.

5.7 Security Testing

Software:

- Make sure that queries to the database are only coming from the predetermined queries necessary to pull the song information
- Make sure user/account information is secure in that it is not possible for users to access other users' accounts

Hardware:

- not applicable, except in maybe a circumstance where a user throws the kalimba to break a window or something, but that's on the user.

5.8 Results

Ultimately, our product will be able to detect played notes and the application will determine if it is a correct note or not. The software will then send the hardware what note should light up next. This process should be decently easy to test and will largely determine if we have met most of our requirements.

So far two different hardware tests have been conducted. They are both described below.

Hardware Test number 1 (10/25)

A pickup for the kalimba has been tested. A piezoelectric pickup was attached to the kalimba and then subsequently plugged into a guitar amplifier. The guitar amplifier has to be turned up to full volume for a soft output sound to be heard; although, the output sound was just as clear and full as the kalimba itself. After some research, it has been determined that a preamplifier is going to have to be added to our design in order for the software to receive a waveform that can actually be used. The issue is that the output impedance of the piezo pickup is high, around 1 Mohm, and the input impedance of most amplifiers is not relatively high, which is why the output on the guitar amp was still soft despite being at max volume.

Hardware test number 2 (11/7)

On 11/7 I tried to measure the voltage signal output of the piezo pickup. To do so, I wrapped wires around the aux cable connected to the pickup (note, I have not tested this aux cable with the guitar amp yet, and I didn't have rubber bands or anything to make sure there was a strong connection between my leads and the aux cable) and interfaced those wires with an LM324 op amp on my bread board, which was set in a typical inverting configuration with an ideal gain of -1E5 V/V. Using a signal from the voltage generator instead of the pickup, I was able to confirm that the amplifying circuit worked as expected . However, even with the 1E5 gain, I was not able to observe a signal on the oscilloscope after plucking a tine. The noise was on the level of 3 to 4 volts peak to peak. Potentially, the signal is still too small to pick apart from the noise even with a gain of 1E5 V/V. Potentially, the connection between the aux cable and the amplifying circuit (haphazardly wrapped wires) was not strong enough to consistently pass signal. Potentially, I have made some other mistake. These are the resources I used to make the circuit so I don't have to look them up later.

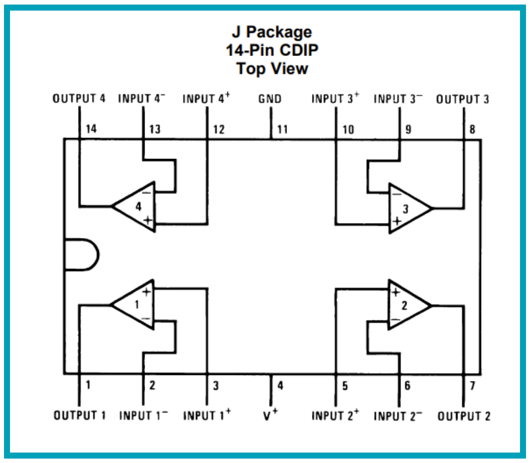


Figure 5.1

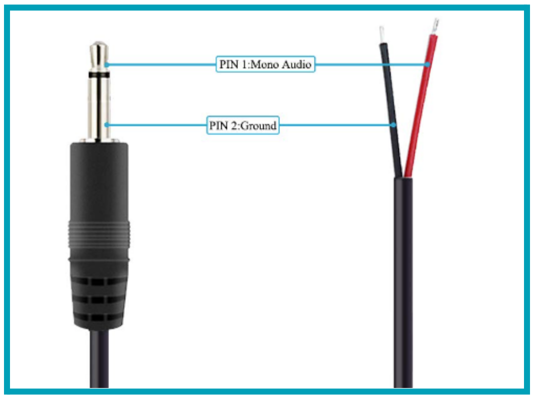


Figure 5.2

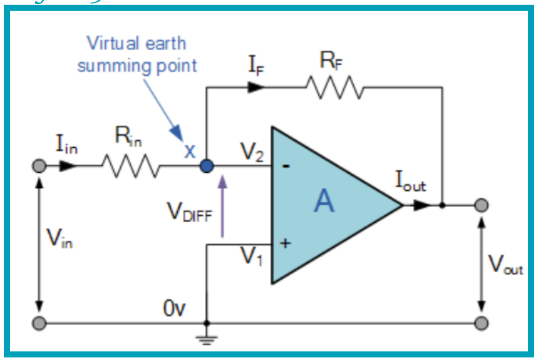


Figure 5.3

$$\text{Gain}\,(\text{Av}) = \frac{V_{out}}{V_{in}} = -\frac{R_f}{R_{in}}$$

Figure 5.4

Hardware Test #3 11/9 Update

We set up note detection testing using the audio pickup and an amplifier circuit using an op-amp as pictured below. With an input resistance of 10Ω and a feedback resistance of 10MΩ the amplifying circuit was able to increase the voltage of the input signal from the pick up jack to be accurately measured for note detection.

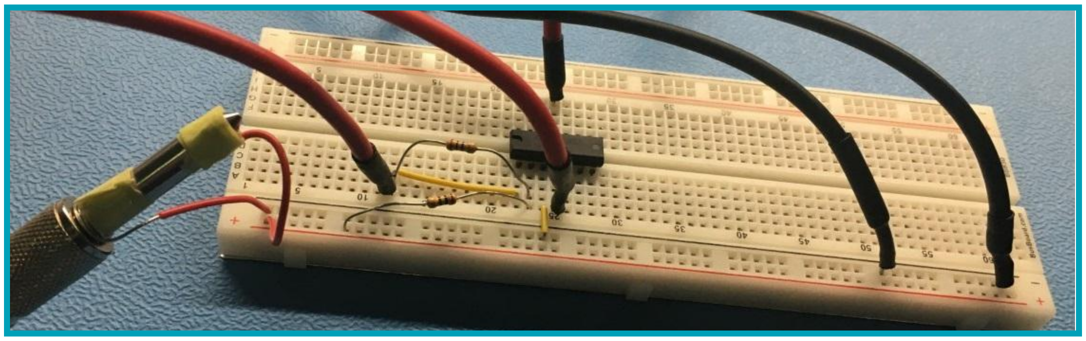


Figure 5.5

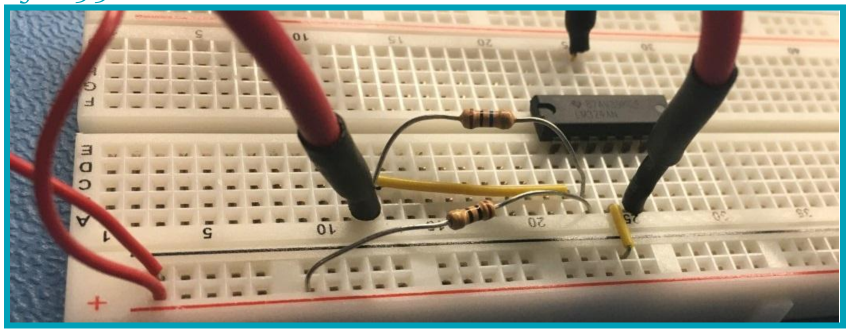


Figure 5.6

For the first test, we designed an op amp circuit with the standard LM324 op amp in the non inverting configuration with Rf equal to 2.6 Mohms and Rin equal to 10 Rin, so an ideal gain of 2.6E5 V/V. When plucking the tine that corresponds to C4, we get the following waveform.

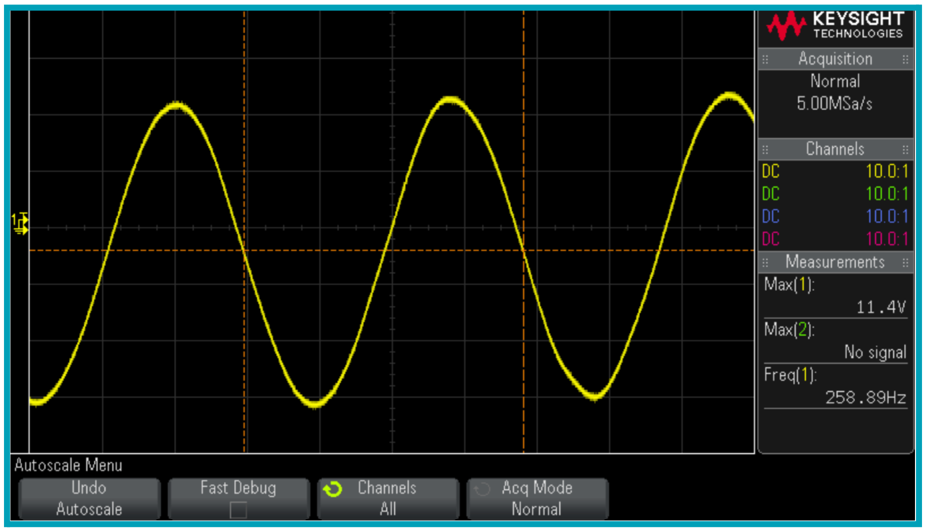


Figure 5.7

C4 should be 262 Hz, and the frequency of the scope reads 258 Hz, only a little out of tune.

When we pluck C4 and C6 simultaneously (frequency of 1047 Hz) we get a resulting frequency of 262 Hz, the frequency of C4, superimposed with the frequency of C6 (1047 hz) just as expected.

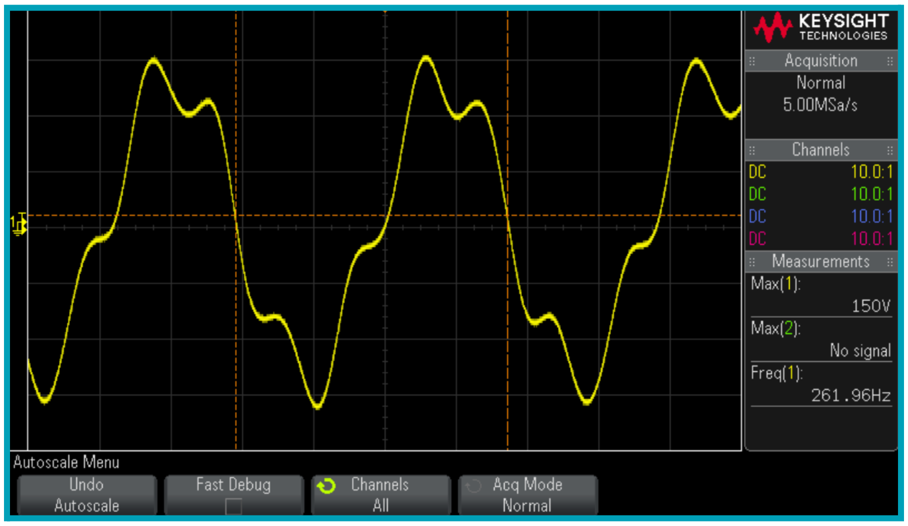
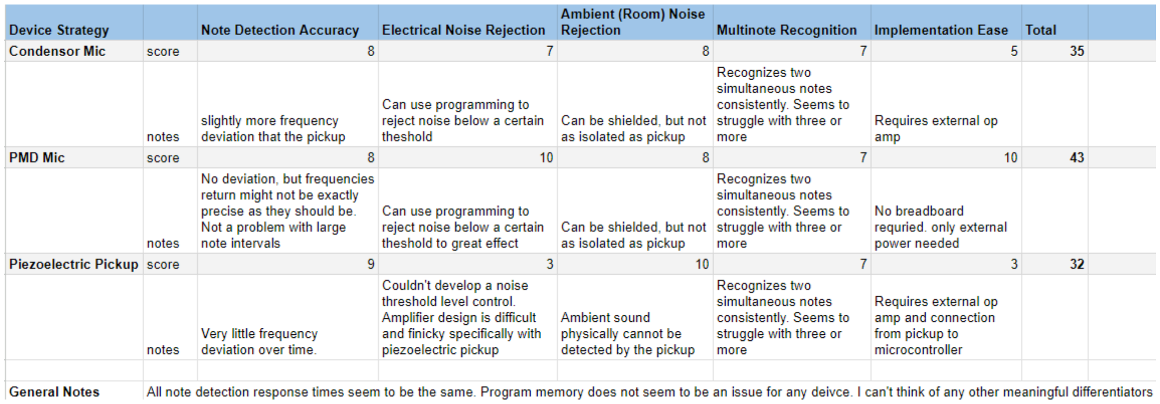


Figure 5.8

Using the multimeter to measure the resistance of pickup, we were getting values that fluctuated from .5 Gohm to 1 Gohm. After a quick google search, I found that multimeter resistance is measured with either constant current or constant voltage. I do not know how the specific multimeter in the TLA was operating, but the resistance of the pickup was fluctuating quite a lot. (over an approximate range of 500 Mohms). This fluctuation is probably due to the fact that the piezoelectric pickup is not technically a conductor, so trying to force current through it or place a voltage across it will not create behavior similar to measuring a resistor. Long story short, the input resistance of the Kalimba is quite high but not known exactly.

Final Hardware Test (3/29)

Once all the potential methods of note detection were implemented in a usable fashion on their own, each method was run through a series of tests to determine which option would best serve the needs of the project. The decision matrix showing the results of the test is shown below. The PDM microphone was determined to be the best option for the job.



Device Strategy		Note Detection Accuracy	Electrical Noise Rejection	Ambient (Room) Noise Rejection	Multinote Recognition	Implementation Ease	Total	
Condensor Mic	score	8	7	8	7	5	35	
	notes	slightly more frequency deviation that the pickup	Can use programming to reject noise below a certain theshold	Can be shielded, but not as isolated as pickup	Recognizes two simultaneous notes consistently. Seems to struggle with three or more	Requires external op amp		
PMD Mic	score	8	10	8	7	10	43	
	notes	No deviation, but frequencies return might not be exactly precise as they should be. Not a problem with large note intervals	Can use programming to reject noise below a certain theshold to great effect	Can be shielded, but not as isolated as pickup	Recognizes two simultaneous notes consistently. Seems to struggle with three or more	No breadboard requried. only external power needed		
Piezoelectric Pickup	score	9	3	10	7	3	32	
	notes	Very little frequency deviation over time.	Couldn't develop a noise threshold level control. Amplifier design is difficult and finicky specifically with piezoelectric pickup	Ambient sound physically cannot be detected by the pickup	Recognizes two simultaneous notes consistently. Seems to struggle with three or more	Requires external op amp and connection from pickup to microcontroller		
General Notes	All note detection response times seem to be the same. Program memory does not seem to be an issue for any deivce. I can't think of any other meaningful differentiators							

Figure 5.9

6 - Implementation And Project Evolution

Microphone/Note Detection:

As discussed in section 5.8, a decision matrix was employed to determine the most effective note detection option to which the result was the pdm onboard microphone. While initially the thought was that the piezoelectric pickup would be the most desirable option for its extreme accuracy and supreme noise rejection ability, the pdm microphone was eventually chosen as the winner mostly due to the fact it was the easiest to physically realize (no amplification necessary) and its accuracy and noise rejection were only marginally worse than the piezoelectric pickup.

LED implementation:

Skinny LED arrays from Adafruit were chosen for their easy individual addressability and customizability in brightness and color. They had to be cut into sections and soldered together again to replicate the tine pattern on the kalimba. This implementation worked better than trying to use and mount individual LEDS or using an LED/Fiber optic system.

Bluetooth Low Energy Software Application Side:

The Bluetooth Low Energy (BLE) implementation was done through the react-native-ble-plx library. The connection is initiated by the user by clicking the 'connect' button on the 'play song screen'. The BLE manager then scans for the kalimba device and starts a listener function for the receive response characteristic which alerts the app when the user plays the correct note. The user can start a song using the 'play song' button which updates the mode of the kalimba to 1 and sends the first note. Once the receive response characteristic is notified that the user played the correct note, the application sends the next note until we reach the end of the song.

Bluetooth Low Energy Hardware Side:

One part that we spent the most time on was finding a way for the BLE to properly communicate between the microcontroller and and the application. This was one of the longest processes that we undertook and ended up spending a few months learning about the processes required, the fundamentals behind the technology, and the importance of staying in touch with the central device.

The Seeed Studio XIAO nRF52840 (Sense) microcontroller has a BLE module integrated onto the board located near the PDM microphone at the bottom of the Seeed Xiao. We would then use the ArduinoBLE library to call functions from throughout our software for integrating Bluetooth functions into our code. Near the top of the main code, before the setup function, we define the BLE service, characteristics, and characteristic descriptions along with an appropriate UUID that would uniquely distinguish each of them. There is a single service that represents the single kalimba mount, and multiple characteristics used for exchanging information between the two devices. Lastly, we needed to place BLE.poll functions throughout our code to update the BLE radio events for the specified BLE device so that the connection and/or any updates can be handled.

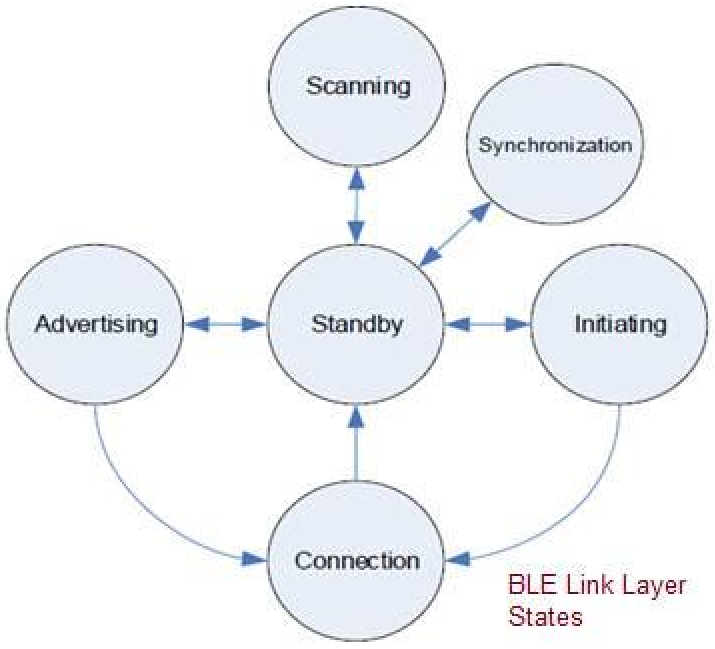


Figure 6.0 (Bluetooth States)

Database:

We looked into several different options for how to store our database. We ended up settling on Firebase due to several reasons. A big reason was that Firebase is a top industry standard for database management and because of this it has lots of good documentation and tutorials. We realized we wanted to choose technology with easy access to items to learn the technology, so the fact that Firebase is very popular and has lots of documentation made it a good choice for us to learn. In addition, Firebase is very easily integrated into React Native and can be deployed to both android and IOS which made it a good choice as well. We also found in our research that it has a quicker learning curve compared to something like AWS.

To implement this database within our project we had to follow several different steps. We first had to setup our database within firebase.google.com and fill our tables with some actual data that would store our songs. Each song was given a value for song id, song title, a favorite boolean and a string array that contained each individual note. Once we had this part set up, we had to figure out a way to connect this to our frontend portion of the app. To do this, created a firebase config file that set up our Twinkle Tines database within our app. We then used several techniques to query the database. In the library and favorites page, we queried the database for all the different song titles and had them show up in our songs panel. We also queried the individual notes array when the user chose a song, and parsed the array into individual notes which were then sent to the kalimba.

User Interface:

The user interface has stayed very consistent since our original design. We originally created the design using Figma where we came up with a theme and a flow of screens. We knew we wanted to keep a common and simple theme to make it easy for the user to navigate and use. Once we settled on a react native application we implemented our design using typescript.

Hardware Mount:

The device mount was created using Autodesk Inventor, a 3D design program. This was used to model each mount design allowing for easier testing. Makerbot printing was used to model and print each mount. The LED display can be attached to the top of the device while the battery and microcontroller can be housed in two boxes on the side. Once these pieces have been connected the mount can be slid onto any kalimba, the four supports will slot at the back of the device and around the bridge. We recommend the use of rubber bands to fully secure the mount to your kalimba and to reduce the resonance disruption that can occur between the plastic of the mount and the body of the instrument.

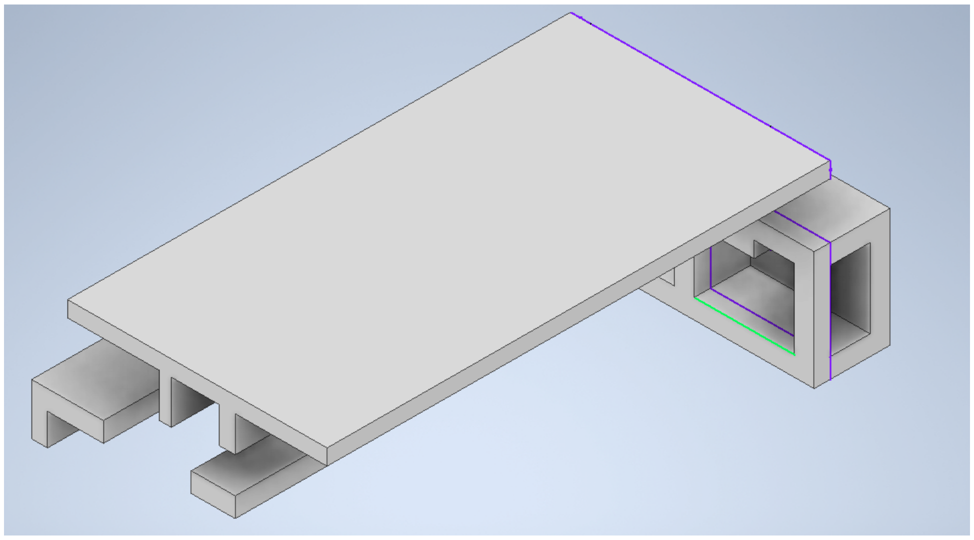


Figure 6.1

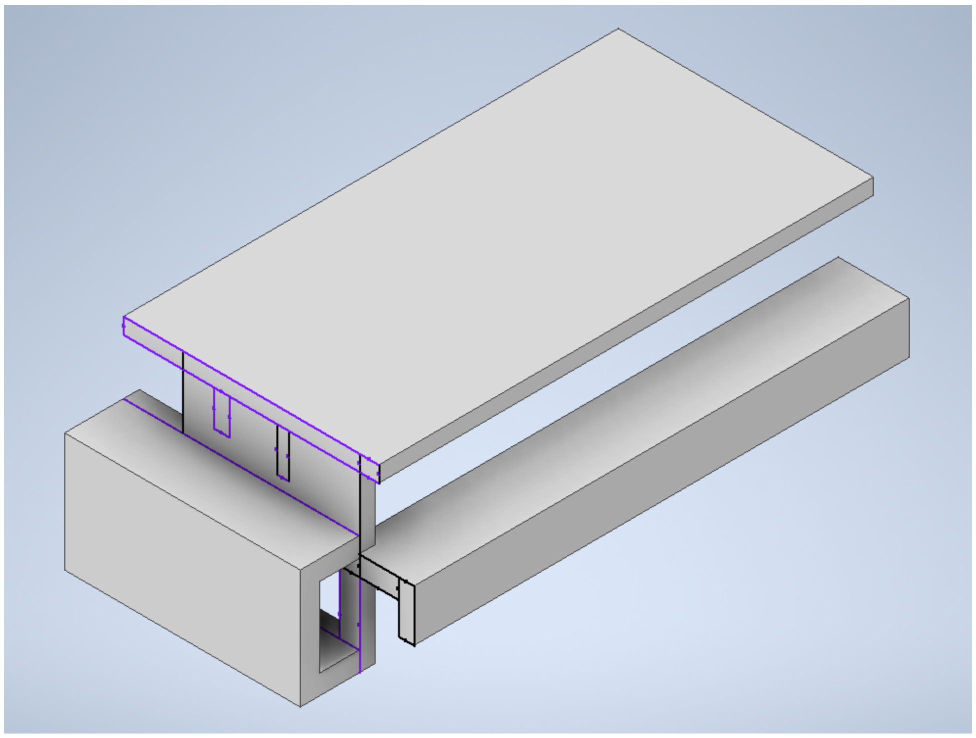


Figure 6.2

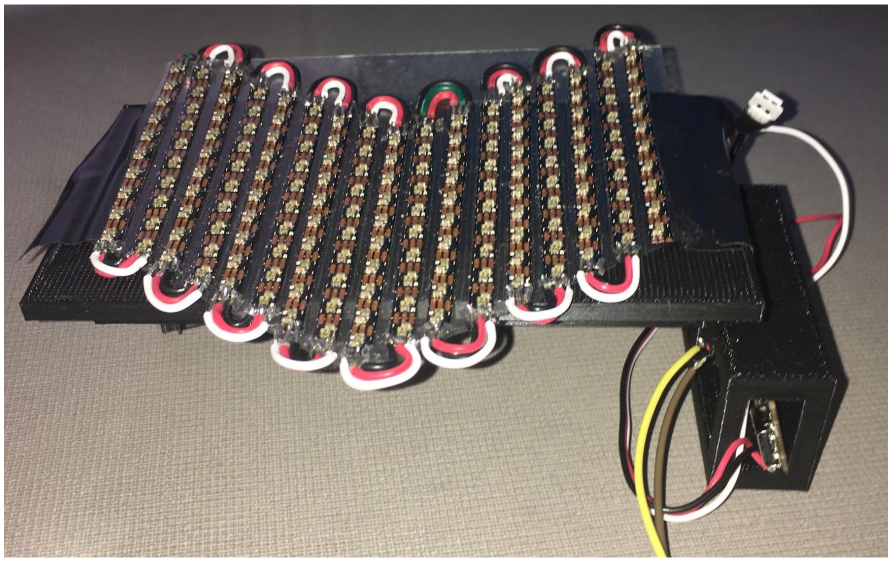


Figure 6.3

7 - Professional Responsibility

This discussion is with respect to the paper titled “Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment”, *International Journal of Engineering Education* Vol. 28, No. 2, pp. 416–424, 2012

7.1 Areas of Responsibility

Area of Responsibility	Definition	NSPE Canon	Our Description	How the ‘Definition’ and ‘NSPE Canon’ Columns Differ
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence.	Perform services only in areas of their competence; Avoid deceptive acts	In order to be able to work ethically, a professional should make sure that they are delivering work and performing to the standards required. To do this, it is important that	The ‘Definition’ seems to be defining examples for what attributes fit into work competence. It also states in general that no matter what the professional is working on they should perform

			the professional sticks to knowledge areas they are comfortable with.	competently. The ‘NSPE Canon’ seems to go one step further and states that the professional should actually only work in their knowledge areas to avoid incompetence.
Financial Responsibility	Deliver products and services of realizable value and at reasonable costs.	Act for each employer or client as faithful agents or trustees.	The professional should never charge more than their service is worth and the customer should not worry that the professional is charging them incorrectly.	The ‘Definition’ says that the professional should judge the cost of their product/service responsibility and should not overcharge but the ‘NSPE Canon’ says that the customer should be able to trust the professional's cost.
Communication Honesty	Report work truthfully, without deception, and understandable to stakeholders.	Issue public statements only in an objective and truthful manner; Avoid deceptive acts.	All communication between the professional and external recipients should be truthful.	The ‘Definition’ talks about reporting work truthfully but the ‘NSPE Canon’ says that any public statements should be truthful. Therefore, one talks about reporting work and the other about public statements.
Health, Safety, and Well-Being	Minimize risks to safety, health, and well-being of stakeholders.	Hold paramount the safety, health, and welfare of the public.	All of the work done and the finished product should not disrupt the health, safety, and well-being of anyone.	The ‘Definition’ column specifies the health and safety of stakeholders but the NSPE says to keep in mind the health and safety of

				the public.
Property Ownership	Respect property, ideas, and information of client and others.	Act for each employer or client as faithful agents or trustees.	Don't damage or take the physical or intellectual property of others.	The first definition is more specific and lists property examples to be mindful of while the NSPE generically states that the customer should trust the professional.
Sustainability	Protect environment and natural resource locally and globally.	N/A	The work and final product should not disrupt the environment.	N/A
Social Responsibility	Produce products and services that benefit society and communities.	Conduct themselves honorably, responsibly, ethically and lawfully so as to enhance the honor, reputation, and usefulness of the profession.	All work and the final product should not disrupt any communities and each individual employee should be respectful of others.	The first definition is very general and says that the work should not disrupt society or other communities but the NSPE is specific to the individual and how the employee should conduct themselves.

Figure 7.1

7.2 Project Specific Professional Responsibility Areas

Area of Responsibility	Definition	NSPE Canon	Important to our Project?	How well our team is performing
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence.	Perform services only in areas of their competence; Avoid deceptive acts	**Yes** Our product can not affect the overall structure of the kalimba or the kalimba will not be able to be played.	**High**. A few group members have experience with a kalimba already which has greatly improved our work competence so far during our research stage.
Financial Responsibility	Deliver products and services of realizable value and at reasonable costs.	Act for each employer or client as faithful agents or trustees.	**Yes** This will hopefully not be much of a concern because our product will hopefully not be too expensive to produce. It is also less of a concern since it is more 'for fun'.	**N/A** We are only just beginning to look at components for our design so we haven't discussed much of the financial aspects.
Communication Honesty	Report work truthfully, without deception, and are understandable to stakeholders.	Issue public statements only in an objective and truthful manner; Avoid deceptive acts.	**Somewhat** We don't think this will be too much of a problem for our group. We have done well so far of documenting our project correctly and I think it will be very evident if our product works in the end.	**High**. We feel like the team has done a great job so far of filling out our design documents and reporting our work there. We also have kept track of accomplishments and milestones through another google doc.
Health, Safety, and Well-Being	Minimize risks to safety, health, and well-being of stakeholders.	Hold paramount the safety, health, and welfare of the public.	**Yes** This also shouldn't be much of a concern because our project will only affect the health, safety, and	**N/A** We haven't been producing any physical products yet an nothing has really come up safety-wise.

			well-being of others by reducing stress.	
Property Ownership	Respect property, ideas, and information of clien and others.	Act for each employer or client as faithful agents or trustees.	**Yes** As it is designed currently, our product will attach to a pre-existing kalimba so we will need to be careful that it does not damage the kalimba.	**Medium.** We haven't had to work muc with the kalimbas yet so we ranked this as a medium but so far nothing has been damaged
Sustainability	Protect environme and natural resour locally and globally	Engineers are encouraged to adhere to the principles of sustainable development1 in order to protect the environment for future generations.	**No** Our product should not have much sustainability impact. It is very compact and will require minimal components.	**N/A** We also haven't had much to do with this yet and we probably won't impact sustainability through the project much.
Social Responsibility	Produce products and services that benefit society and communities.	Conduct themselves honorably, responsibly, ethically, and lawfully so as to enhance the honor, reputation, and usefulness of the profession.	**No** We don't believe our product will have much of a social impact either.	**High.** The goal of our product is to hopefully allow anyone to play the kalimba and we thin we are achieving that throug our designs so far.

Figure 7.2

7.3 Most Applicable Professional Responsibility Area

Work Competence

Our project has to be completed as close as possible to our specifications because any error could cause the kalimba to not work properly. The LEDs will also need to line up to the tines and the mobile app will need to correctly store the current note. Any error in these capabilities will break the core functionality of our project.

8 - Closing Material

8.1 Concluding Thoughts

Despite doubts and problems encountered during the process and along the way, the project has reached a conclusion to be proud of. The prototype implemented meets the initial goals set out in the beginning and addresses the user needs originally identified.

Furthermore, this project has a lot of potential for future growth.

8.2 Team Contract

Team Members:

1) Mesa Hassel   2) Eileen Hillier

3) Kaitlyn Nolting   4) Stuart Pearson

5) Andrew Adams   6) Daniel Duerr

7) Isaac Vrba

Team Procedures

1. Day, time, and location (face-to-face or virtual) for regular team meetings:
   a. Mondays at 8:10pm at the library, 3rd floor.
   b. Thursday during 491 class time

2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):
   a. Preferred Method of Communication:
      i. Discord
      ii. Outlook
   b. Reminders
      i. Meeting Reminders: two hours before
      ii. Deadline Reminders: two days before
   c. Issue Resolution Method
      i. Upon first concern, dm the person to discuss the issue immediately
      ii. If a solution is not immediately reached, bring it up to the team.
      iii. If a solution is still not reached, bring it up to the advisor (Mani Mina)/Professor Fila
   d. Scheduling
      i. when2meet to schedule times that work for majority of the team
      ii. scheduling meetings with advisor, choose 3 times that work for the advisor and select the time majority of the team can make it to

3. Decision-making policy (e.g., consensus, majority vote):
   a. Consensus for major project decisions

b. For sub-groups that are focusing on smaller decisions, they have the liberty to make smaller decisions as long as they are filling in the rest of the group with their decisions.
c. If unable to come to a team decision, ask the question to the advisor or other specialist for input.

4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be
shared/archived):
    a. One person on team will record our minutes each meeting
    b. They will be shared via a Google drive document called Meeting Notes

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:
    a. Expect each team member to make each meeting unless a notice is given 2 days in advance.
    b. If not able to make the meeting, be sure to catch up on what the team went over by reading through the minutes and asking questions about things that were discussed.
    c. Expect each team member to attend each Thursday lecture unless a notice is given 2 days in advance

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:
    a. You are to complete all team assignments, timelines, and deadlines before the decided team deadline.
        i. If any of the above is needed to change, you must communicate to the group for a goal re-evaluation.

3. Expected level of communication with other team members:
    a. weekly at team meetings
    b. weekly in class with a quick status update

4. Expected level of commitment to team decisions and tasks:
    a. Team members are expected to meet all deadlines that were set by the group
    b. Team members are not to “go dark” and continuously communicate with each other about any progress.

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):
    a. Hardware half:
        i. Stuart - Client communication point
        ii. Eileen - Hardware
        iii. Isaac - Git Expert
    b. Software half:
        i. Andrew - User Interface

        ii. Mesa - Sub group meeting planner
        iii. Kaitlyn - Sub group meeting minutes
        iv. Daniel - Design
    c. **These roles are not final and subject to change as project develops**

2. Strategies for supporting and guiding the work of all team members:

    a. Show up to meetings
        i. Group meetings will be for updating everyone on the sub-team progress and for group decision making
        ii. Sub-Group meetings that are aimed for keeping all the members of the smaller teams in the loop.
    b. Show up to class
    c. Discuss in Discord continuously

3. Strategies for recognizing the contributions of all team members:

    a. A jingle gets played on the Kalimba
    b. Give updates during meetings

Collaboration and Inclusion

1. Skills, expertise, and unique perspectives each team member brings to the team:
    a. Mesa (SE) - Experience using Git, experience using android studios and xcode for both android and ios apps, HTML, Figma app planning/design
    b. Andrew (SE) - Experience using Git, front end and back end internships (React, Java, etc)
    c. Stuart (EE) - Soldering, hardware testing and troubleshooting, cinda creative™
    d. Eileen (EE) - Power systems, hardware systems, signal processing
    e. Kaitlyn (SE) - Experience with Git, Android Studio, Java, HTML, Front-End and Back-End development, Figma,
    f. Daniel (SE) - Experience with Git, Visual Studio, HTML, Java, C, Creative, Unique, Can make buttons, like really cool ones
    g. Isaac (CprE) - Experience using Git, Soldering, android studio, embedded work, C, Java, Front-end

2. Strategies for encouraging and support contributions and ideas from all team members:

    a. Celebrate Success
    b. Learn from failure
    c. Document Successes, all of them inside the Success and Milestone Document

3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will
a team member inform the team that the team environment is obstructing their opportunity or ability to contribute?)
    a. Never you versus me, always us versus the problem
    b. Group discussion (Virtual/Face-Face)
    c. Be there for each other
    d. Overcome challenges as a team

Goal-Setting, Planning, and Execution

1. Team goals for this semester:

    a. Have fun
    b. Have a well ideated plan to shoot for
    c. Start on the build of the project, both hardware and software before end of semester
    d. Create effective, detailed documentation and have it organized
    e. Apply what we learn in class to the project
    f. Learn a new technical skill during the course of the project

2. Strategies for planning and assigning individual and team work:

    a. At every weekly meeting, we are to go over the next sprint period's goals and between the set period a series of smaller sprint goals is to be set to keep all team members freshly updated of what is expected of them.
    b. Starting a team meeting with standups.

3. Strategies for keeping on task:

    a. Issues in Git for individual tasks to see progress
    b. Ask for updates from team members during meetings
    c. Participate regularly in meetings

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?

    a. Reach out and ask what happened that caused non participation
        i. Work out how to stay better on track in the future

2. What will your team do if the infractions continue?

    a. Work with the student to solve the problems preventing participation
    b. Contact advisor or professor to arrange long term consequences

---

a) I participated in formulating the standards, roles, and procedures as stated in this contract.
b) I understand that I am obligated to abide by these terms and conditions.
c) I understand that if I do not abide by these terms and conditions, I will suffer the consequences as stated in this contract.

1)	Mesa Hassel	**DATE**	9/12/2022
2)	Andrew Adams	**DATE**	9/12/2022
3)	Stuart Pearson	**DATE**	9/12/2022
4)	Eileen Hillier	**DATE**	9/12/2022
5)	Kaitlyn Nolting	**DATE**	9/15/2022
6)	Daniel Duerr	**DATE**	9/12/2022
7)	Isaac Vrba	**DATE**	9/12/2022

Appendix

Appendix A: Instruction manual

To get to a point where you are able to use the kalimba mount device, a react-native environment needs to be created so that the application can be loaded and able to communicate with the Seeed Xiao microcontroller. The steps for starting an installation are as follows:

1. Set up react-native environment, follow the instructions on the official React-Native website: https://reactnative.dev/docs/environment-setup
2. Clone Twinkle Tines Repo from the ISU GitLab: https://git.ece.iastate.edu/sd/sdmay23-30.git
3. Open two terminals, in one run npx react-native start, in the other run npx react-native run-android

Once the application loads, you will see the title screen and have the option of going to the song library, view your favorite songs, and/or going to the settings. Within the library page is a list of all the songs that have been added to the database and are available to be played. In the favorites page, only the songs that the user has favorited will be displayed. Again, you can scroll through this list to choose a song. As of this time, the settings page is blank and has not been implemented yet.

Before any song can be played, we need to connect to the microcontroller via Bluetooth. Power on the microcontroller either by plugging in a battery to the two open wires located near the battery pocket, or by powering it with the USB Type-C cable that connects to the Xiao located at the bottom right of the mount. Once it is powered, the microcontroller will start searching for a central bluetooth device.

When the application connects to the mount device, the user will have the option to either play the song or favorite the song. If the user favorites the song it will updates the database and will be available to be selected on the favorite screen.If the user decides to play the song, the notes string will be pulled from the database and one note will be sent to the microcontroller at a time, updating only if the user plays the right key. This loop stops once we have reached the end of the song string.

The current version of the project code currently only supports full-tine lighting when it receives notes from the application. Planned within any future work on this project, we would like to see the implementation of a "Waterfall Mode". This will accept an array of input notes in case multiple notes need to be played, and send the note down the LED canvas as the time it will be played approaches.

When the application is done sending notes for any given song, it will turn off all the LEDs on the mount and wait for the user to either select another song or to power off the device.

Appendix B: Scrapped Designs

1. USB Type-C as the sole connection from a phone to the device
    a. We opted for a bluetooth connection to transmit data between the microcontoller and Twinkle Tines Applicaiton.
    b. This worked a lot better for the design of our mount and convenience for the user.
    c. When we decided to use bluetooth, we also wanted to find a way to run the device with a battery so that we were not stuck with a tethered device.
2. Piezoelectric Pickup and Condenser microphone for the analog signal detection
    a. The pdm microphone was judged to work the best of the three options. This hardware element was built into the microcontroller and was the most reliable for detecting the notes played as determined by our decision matrix.
3. Lighting using fiber optics
    a. When we were problem solving for how to best notify the user about incoming notes we considered using fiber optic wire since it was cheaper, did not require wiring on the bottom, and was the most durable option we found.
    b. We ended up scrapping the idea after we received some to test with. We found that when we used them we would be sacrificing the ability to run waterfall code in the future.
    c. To supply a light source to the fiber optics would require 17 Optical Transmitters and those take up more space and to connect the thin optic was going to be very challenging for our design.
4. 3D Printed mount designs
    a. Chevron shaped top base to support LED display above kalimba tines
    b. Moving mount pieces to allow for complex size adjustments for different kalimbas
    c. Secondary model piece to support the device and ensure a tight connection to the kalimba
    d. Adjusted designs to improve measurements
    e. Previous designs that miss microcontroller/battery storage and additional support structures