ECE1778 Final Report

Flexinome

Specialist: Aaron Chow

Programmer: Hongyi Yang, Lexi Zhu

Word count: 2218 Penalty: 0

1. Introduction

Motivation

Metronomes are useful simple tools, however, they do not adapt to the composer's intentions, such as pauses, slowing down, and speeding up, not to mention complex time signatures and rhythms. Turning pages has always been a compromised experience; Bluetooth page-turners are awkward and inconvenient while taking hands off the instrument to turn pages leads to missed notes and can lead to disastrous outcomes.

Goal

The goal of our app is to help musicians in performance and practice by having an adaptable metronome that is programmed by the user to follow the tempo and time signature indications of any piece, as well as a clever and efficient way to turn pages through gesture and time.

2. Statement of Functionality

The following sections describe the functionalities of the app along with corresponding screenshots.

Metronome

The metronome feature simulates a physical metronome. After the main screen is presented (Figure 1), the user can tap on the 'Metronome' button to enter the metronome screen (Figure 2). On the metronome screen, the tempo of the metronome is displayed at the center as beats per minute (BPM). Next to the tempo is the time signature value which is displayed as a fractional number. By tapping on the Start button, the metronome will start to play beats sound.

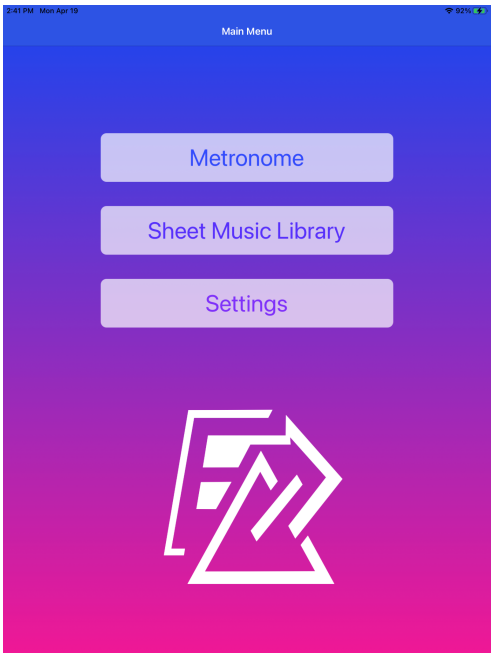

Figure 1. Main Screen

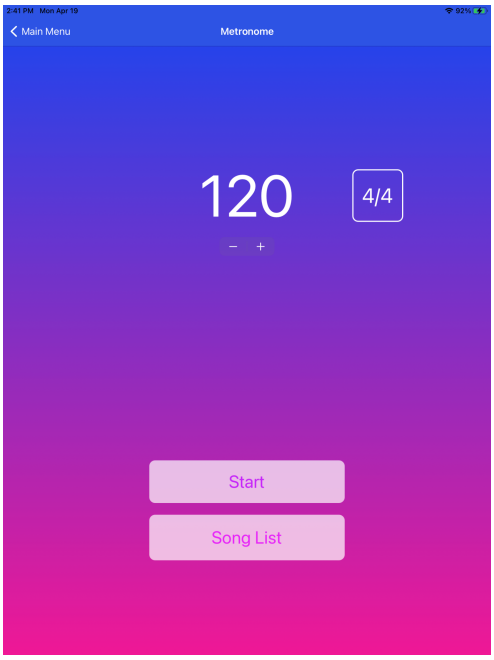

Figure 2. Metronome Screen

The user can change the tempo to make the metronome faster or slower. The time signature value can also be modified to change the number of ticks the metronome plays. When the user taps on the time signature on the screen, the time signature screen will pop up (Figure 3). By pressing the ‘set’ button, the app goes back to the metronome screen and updates the time signature.

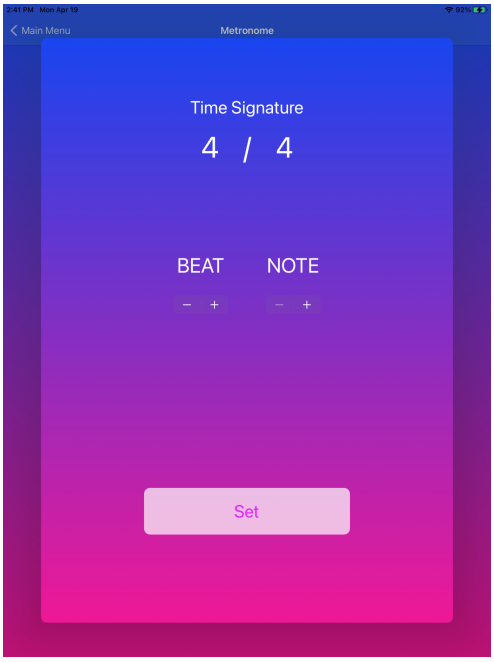

Figure 3. Time Signature Screen

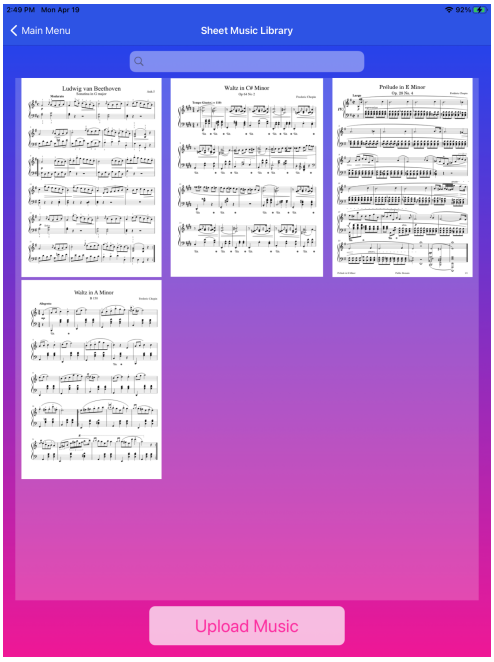

Figure 4. Sheet Music Library

Music Score Reader

The music score reader allows the user to upload and view music scores in pdf format. The user can tap on the ‘Sheet Music Library’ button to enter the sheet music library screen (Figure 4). This screen shows all the music scores (pdf files) that the user has uploaded. By tapping on the ‘Upload Music’ button, the user can upload a pdf file stored on the local storage of the device. The app allows searching for an uploaded music score by its name (Figure 5). By tapping on a thumbnail of the music score, the app will display the file on full screen (Figure 6).

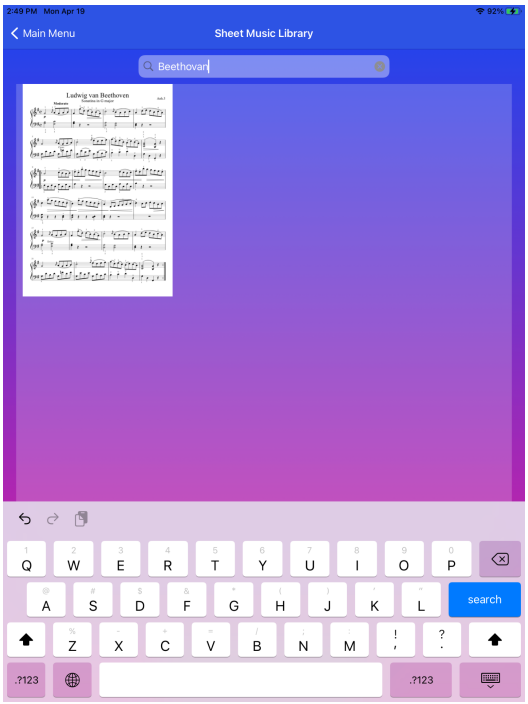

Figure 5. Searching

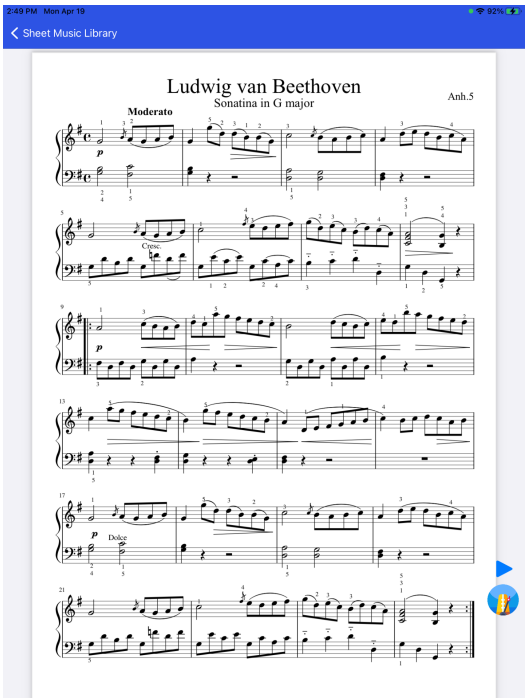

Figure 6. Music Score Reader

Sequencer

The sequencer in the app acts as a programmable metronome. It lets the metronome change its tempo and time signature automatically at a time specified by the user. This feature allows the metronome to adapt to music that changes tempo or time signature in the middle of the song.

Each of the user’s configurations of the sequencer is stored as a ‘song’ in the app. When the user taps on the ‘Song List’ button in the metronome screen shown in Figure 2. The app first

presents the song list screen (Figure 7) where it shows a list of songs. A new song can be added by tapping on the '+' button on the top left corner. Then the sequencer configuration screen will be presented (Figure 8). In our sequencer, a song is divided into many sequences (rows in the table). The user can specify what tempo and time signature he or she wants the metronome to play at a specific bar. For example, if a song changes its tempo from 120 to 130 at bar 5, then the user can set the tempo to 120 from bar 1 to 4 and set the tempo to 130 from bar 5 as demonstrated in Figure 8.

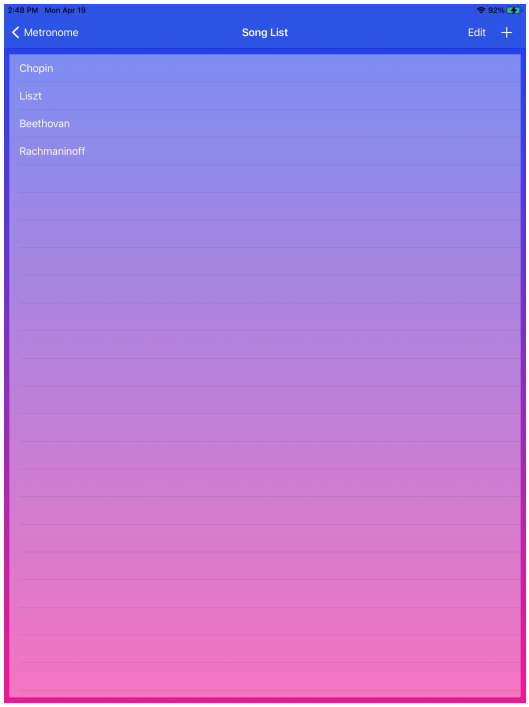

Figure 7. Song List Screen

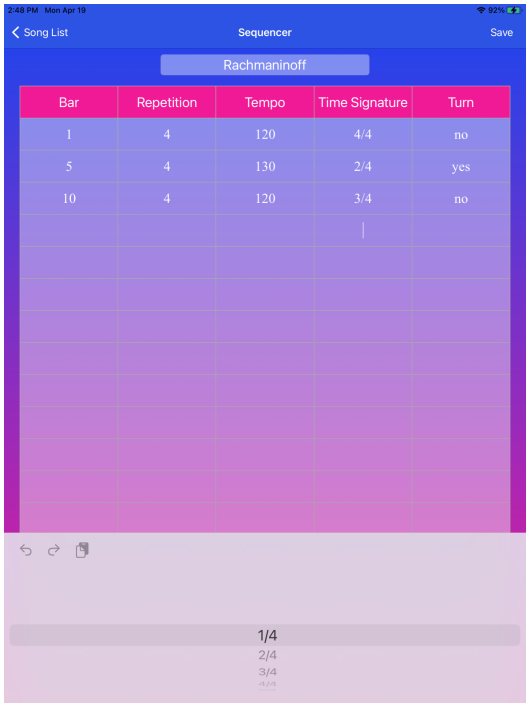

Figure 8. Sequencer Configuration Screen

After the new song is saved, it can be selected and the app goes back to the metronome screen. The metronome will then enter the sequencer mode with the configuration stored on this song(Figure 9). On this screen, an indicator on the center of the screen displays the current bar being played. As the metronome starts playing, once it hits a certain bar, it updates its tempo and time signature based on the configuration (Figure 10).

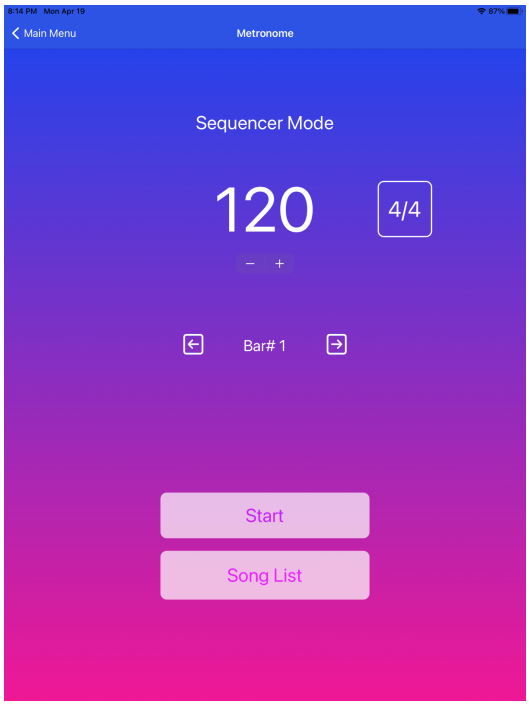

Figure 9. Sequencer Mode

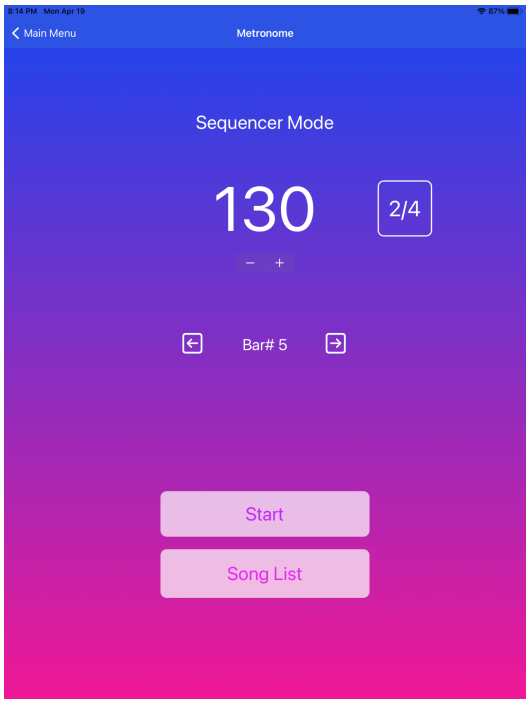

Figure 10. Metronome Changed

Music Score Reader with Metronome and Sequencer

The music score reader can work together with the metronome and the sequencer which allows the user to practice along with the metronome while viewing a music score. The user can access the metronome from the music score reader by tapping on the button at the bottom right corner (Figure 6). The song of the sequencer can be loaded in the aforementioned way. Once completed, the ‘set’ button (Figure 11) takes the user back to the music score reader page (Figure 12). The user can now start the sequencer or the metronome in the music score reader screen.

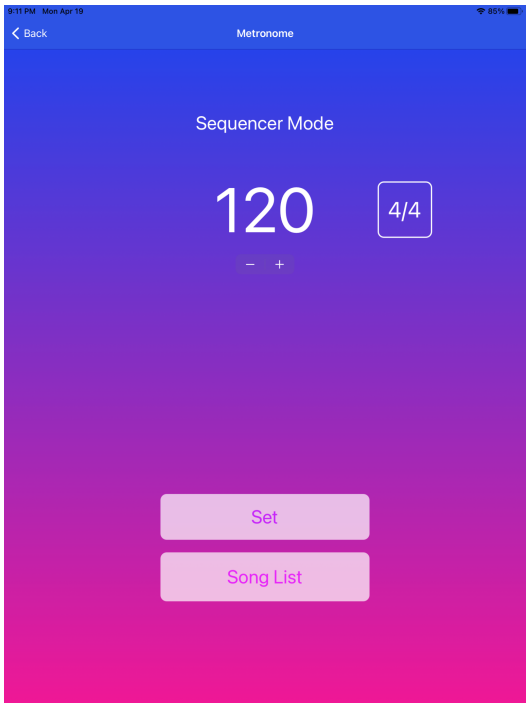

Figure 11. Setting the Sequencer

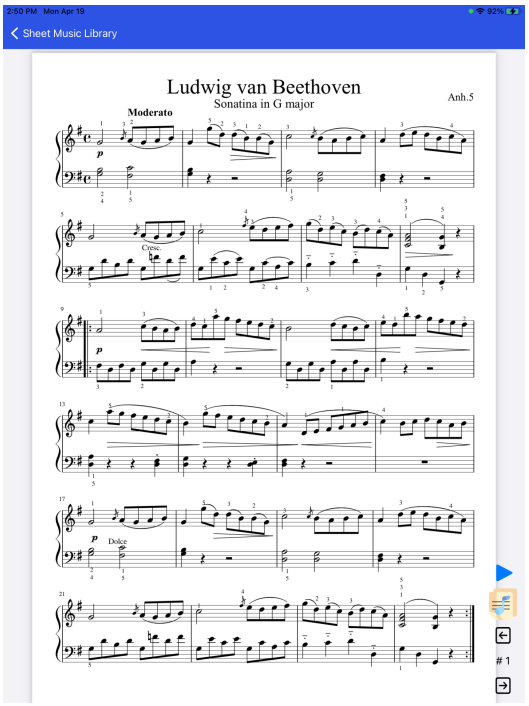

Figure 12. Sequencer with Score Reader

Automatic Page Turning

The user can configure automatic page turning on the sequencer screen (Figure 7). At the last column of the table, the user only needs to choose whether or not the page will be turned at the end of that sequence. Then when using the music score reader with the sequencer, the app will flip the page for the user at the specified time.

Page Turning by Head Gestures

The head gesture recognition feature allows the user to turn pages with head tilts so users can turn pages without their hands leaving the music instruments. The user can change the sensitivity of gesture recognition on the settings screen (Figure 13).

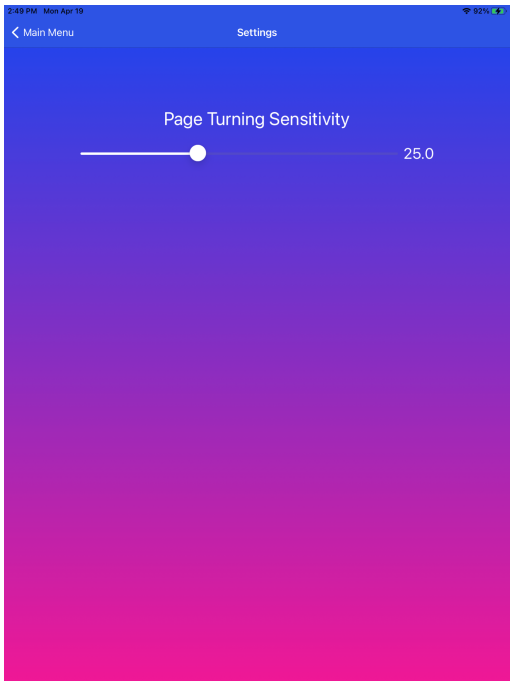

Figure 13. Head Gesture Sensitivity Setting

3. Design

The software structure of the app is shown in Figure 14. There are three main components in the app: the Metronome Audio Engine, the Sequencer Algorithm, and the Music Score Reader.

- **Metronome Audio Engine**: It is built using the AudioKit API [1]. It takes in user inputs such as tempo and time signature, and then generates beats and outputs the ticking sound.
- **Sequencer Algorithm**: It takes in a sequence of tempos, time signatures, and page turning positions as the inputs. This information will be stored as a JSON file on the user's device. The algorithm then loads a file and calculates the correct timing for tempo/ time signature updates by counting the number of beats played. It utilizes a callback function to update the Metronome Audio Engine and the UI. It can also trigger a page turn while it is working together with the Music Score Reader.
- **Music Score Reader**: PDF files can be uploaded here and stored locally on the user's device. It also uses a head gesture recognition algorithm from Google's Firebase ML kit

Face Detection API [2] to identify head gestures. When a head gesture is detected using the user's front-facing camera, the algorithm triggers a page turn on the pdf file.

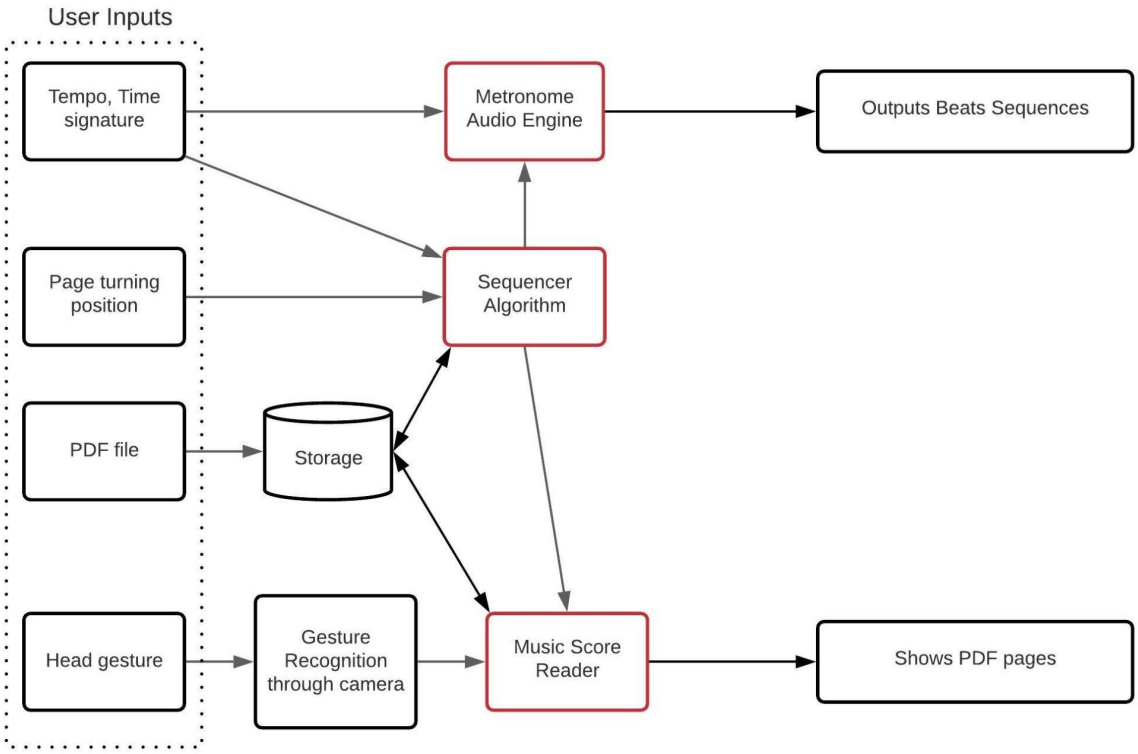

Figure 14. Software Structure

4. Reflection

Agile workflow

One useful development practice we learned and plan to adopt in the future is the agile development workflow. We learned this by treating the two week spirals as sprints and dividing features up based on importance. This made delegating work very easy and efficient and it made modifying, adding, or removing functions very straightforward while we were developing the app.

Do not reinvent the wheel

As we were creating the programmable metronome, we learned the important lesson of trying not to reinvent the wheel. Initially we were going to program the metronome from scratch, but this proved to be very challenging and time consuming. We later were able to find a library that

provided the basic metronome features we needed, and just needed to program changing the tempo, which made our task a lot simpler. If we had just looked for open source metronomes to begin with, we would have saved a lot of time and manpower.

Do more research before committing

We also learned to research more in depth on potential libraries that can be used before committing to one. When developing the pdf upload and page turning functions, we chose to use a specific PDF library, but this library had restrictions that proved to be difficult down the road because it made adding features more complicated than necessary, and thus we learned that we should look at more options before committing to a library.

Importance of user feedback

We also learned that user feedback is very helpful in the debugging process. Having our specialist, Aaron, use the app daily helped us find bugs and ways to improve user experience. Programmers test code in a biased manner because they know how it works and how it should behave. Thus an outside user is needed to use the product for purposes it may not be designed for to help find bugs or give suggestions to help improve the app.

Musical terms

Lastly, because we were working on a music app, we learned many music terms along the process. For example, the tempo is the speed at which a passage of music should be played. A time signature specifies how many notes are in a bar, and which note signifies a beat. In a time signature, there are two numbers. The top number tells us how many beats to count, and the bottom number tells us what kind of note to count, or how many beats are in a bar.

5. Individual Contribution

Lexi - Programmer:

- UI
- Implemented music sheet library and music score reader
- Added song searching functionality
- Implemented facial detection page turner and head tilt sensitivity

- Implemented sequencer configuration
- Implemented song list screen

Hongyi Yang - Programmer:

- Implemented the metronome feature using AudioKit
- Implemented the sequencer algorithm
- Integrated the sequencer and the metronome with the music score reader
- UI

Aaron - Specialist

- Bug testing
- Icon design
- App research

6. Specialist Context

Flexinome was conceived to address two problems that plague classical musicians, namely a customizable metronome that is capable of following a composer's intention, and a page turner that understands the player's position in the score and automatically turns. Through a uniquely conceived collaboration of the two features, it creates a tool that is broadly applicable to musical performances, as well as usage in the practice room.

The flexible metronome opens up many possibilities for new music performances, where composers constantly experiment with the manipulation of time. It greatly eases a musician's practice by being able to instantly and reliably change meter and tempo, and therefore follow the composer's intentions more precisely and efficiently. In older music (i.e. music composed pre 20th century), composers indicate accelerations (accelerando) and slow downs (ritardando), as well as abrupt tempo changes, and this can be easily accommodated by the flexible metronome feature.

Page turning has long been a struggle for the musician. Many musicians wish for an app that would listen to them play, follow them on the score and automatically turn pages. While the

technology is starting to become available to achieve this (e.g. Tido and eNote), subscriptions are required for such apps, and proprietary files are needed for the application to operate. With the vastness and variety in music, such limitations prohibit widespread adoption. Rather than figuring which note you are on in the score, Flexinome knows which beat you are on the score through the programmable metronome and will turn pages once you reach a certain beat. Finally, facial expression detection allows for greater flexibility in correcting the position when the performance does not proceed as planned. The universality of this method, along with the use of widely available pdfs increases accessibility and will encourage widespread adoption.

The achievements of Flexinome will aid in pedagogy, where meter and time changes can be changed within the app and clarify any confusion that can happen when using a standard rigid metronome. Other benefits include the ability to enlarge the score and show fewer bars for the visually impaired, while maintaining usability through automatic page turns. Finally, Flexinome opens up the possibility to beat based page turning, which is a much simpler and elegant solution in comparison to the challenges of pitch based page turning, which necessitates the programming every note in the score, and the implementation of pitch recognition and algorithms.

7. Future Work

Music tracking

We would like to be able to highlight the bars on the music sheet so the musician has a better visualization of which part of the music they are on. This highlighted region should follow the player as they go through the music to help the player track where they are on the music score.

Smooth tempo change

Currently when the programmable metronome changes tempo, it happens very abruptly. We would like to add the function of accelerating or slowing down the metronome through a bar as sometimes music scores require this, so that the tempo change can happen more smoothly.

Beat visualization

Beat visualization can help the musicians stay on beat by adding a visual aid in addition to the metronome sounds. Whenever the metronome makes a clicking sound, there sound be a visualizer, such as a blinking light, or an animated metronome on the screen, that the musician can follow in addition to the sound.

Customization

We would also like to add some customization options for the user, such as providing different metronome sounds and offering different beat vidualizers. We also want to allow the automatic page turner to turn pages before the last note, so the musician can prepare for what is to come on the next page.

8. Group Consent

Group Member	Video	Source Code	Final Report
Aaron Chow	Yes	Yes	Yes
Lexi Zhu	Yes	Yes	Yes
Hongyi Yang	Yes	Yes	Yes

References

[1] “AudioKit,” *GitHub*. [Online]. Available: https://github.com/AudioKit
[2] “Firebase Face Detection”, *Google*. [Online]. Available: https://firebase.google.com/docs/ml-kit/detect-faces
[3] “PDFKit”, *Apple*. [Online]. Available: https://developer.apple.com/documentation/pdfkit
[4] “SpreadsheetView”, *GitHub*. [Online]. Available: https://github.com/bannzai/SpreadsheetView