**Tp3 updated design document**

**Project Proposal**

**Project Description**

**Musical Harmonizer**

The application serves as a learning tool for music that will generate harmony lines(in the form of sheet music and sound outputs) based off the melody line played/ sung. Some of the key features include finding the musical key of the user’s recording and also a virtual piano that they can play.

I have seen transposing software that raises/ lowers the notes of the input. I have not seen a harmony generate but the idea stemmed from incorporating those of key finders and musical synthesizers. This is a tool that uses music theory to allow the user to visualize instantly the music they are making.

The four broad categories are the main graphical interface, the analyzers, the transposers, and the music players each stored in separate files. Prior to these steps there are two functions that record the input and play the input back.

Graphical interface

* In the homepage, there is a function to draw the record and playback buttons, the record and play functions from another python file will be called here
* In the KeyAndScale mode, There are three features: the virtual piano( to draw, there are drawWhiteKey, drawBlackKey and drawPiano functions ). There will be some functions that will aid in playing the piano called by both mousePressed and keyPressed.
* For drawing the sheet music, there are a few functions: yPositionCalc, getPositionOfNote, drawLine and drawNode, which aid in drawing the lines and notes depending on the values of frequencies and midis generated by the analysers.
* The keyPressed function calls the various analyser and ply functions from the other python files.

Analyzers

* In a separate python file, there are two functions that analyze and generate the first set of required note values of the song. The names of those functions are analyseWave() and formatMidiListFromPitches()
* In another file, there are some functions called getNoteAndMidi(),filterNewList() and findKey(). These functions format the notes in desirable formats which would allow them to be accessed and processed efficiently by the sound processors.

Transposer

* This is the getHarmonyLine function that generates the harmony based off of the melody.

Music Player

* These functions are in a separate file and they perform the task of playing the notes together and as commanded by the user. The utilize the formatted values generated by the analyzers to play these notes.

All these functions are called by the controller function in the Graphical interface.

The main algorithm is generating the output harmony notes when being provided a wav file. The process of filtering the sound to be analyzed, and using various combinations of midi values and notes along with their time durations (which had to be calculated) all constitute the algorithmic structure. Generating harmonies by transposing could be a simple process, but it is possible that the user can sing or play in any key and the program must change its values according to the key. Therefore, it was necessary to find the key of the song using its midi notes and a chart derived from applying theory of music.

By the Tp1 deadline I planned to implement all of the algorithmic characteristics of the app, which are to generate notes for the harmonies, have the modules interacting smoothly and analyzing the input wav file and format it. I also implemented the sheet music. By the tp2 deadline I plan to have all of the graphical features including the virtual piano and plan for better user interface.

I am backing up all of my folders to the mac iCloud drive.



Module List:

Audio Modules

* pyAudio
* Aubio
* pyGame
* A sound synthesizer(within the computer) called simpleSynth to play the output audio used in pygame

Time sheet

Friday 17th, 2020

* Worked on the inverted color and updated the design of the home page for 2 hours
* Added record button and play button instead of key presses for 1 hour

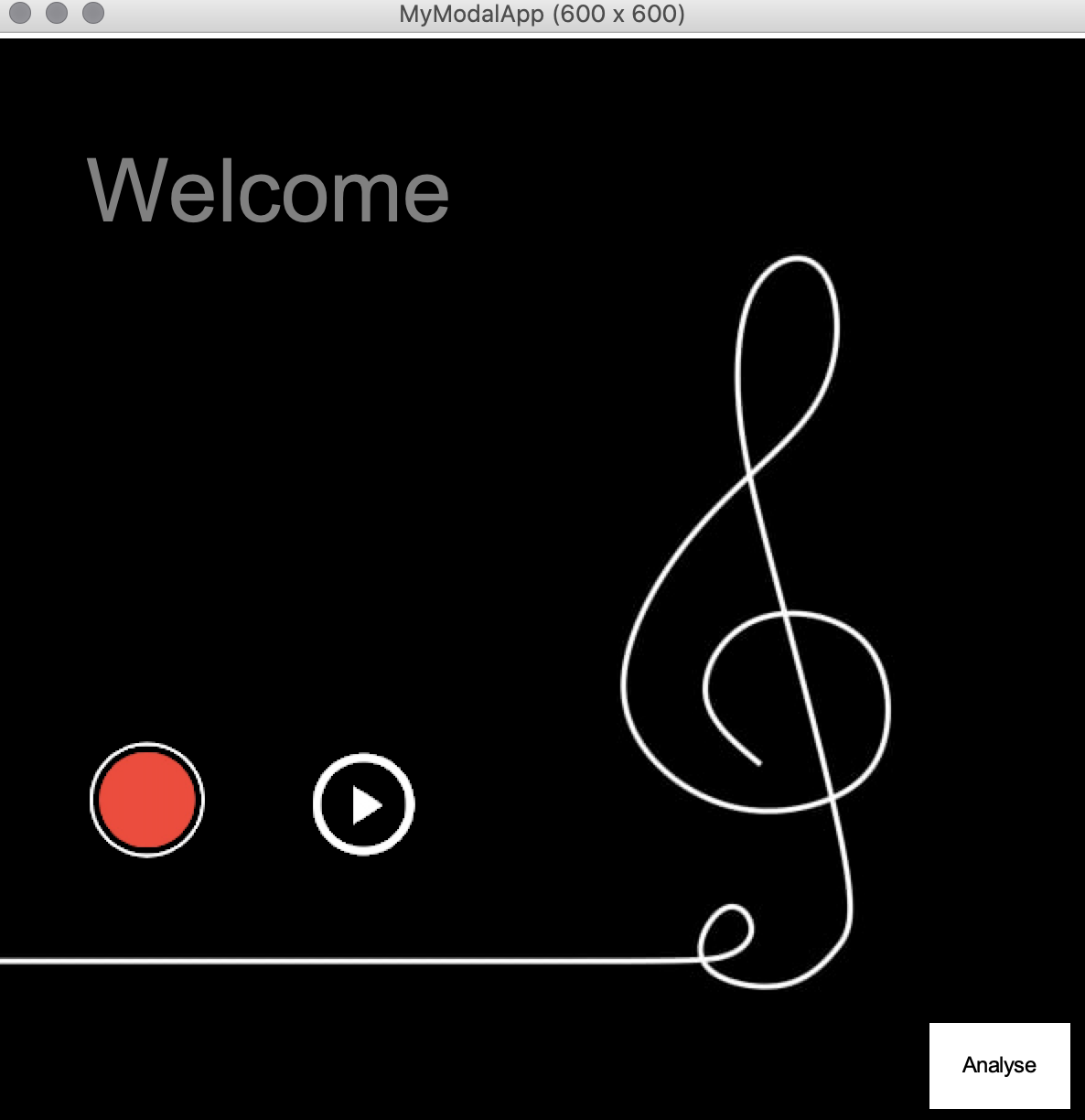
Saturday 18th, 2020

* Found some bugs in the code that calculates key finder function and other musical functions for 7 hours.
* Added staffs to the notes in the sheet music - 2 hours

Monday 20th, 2020

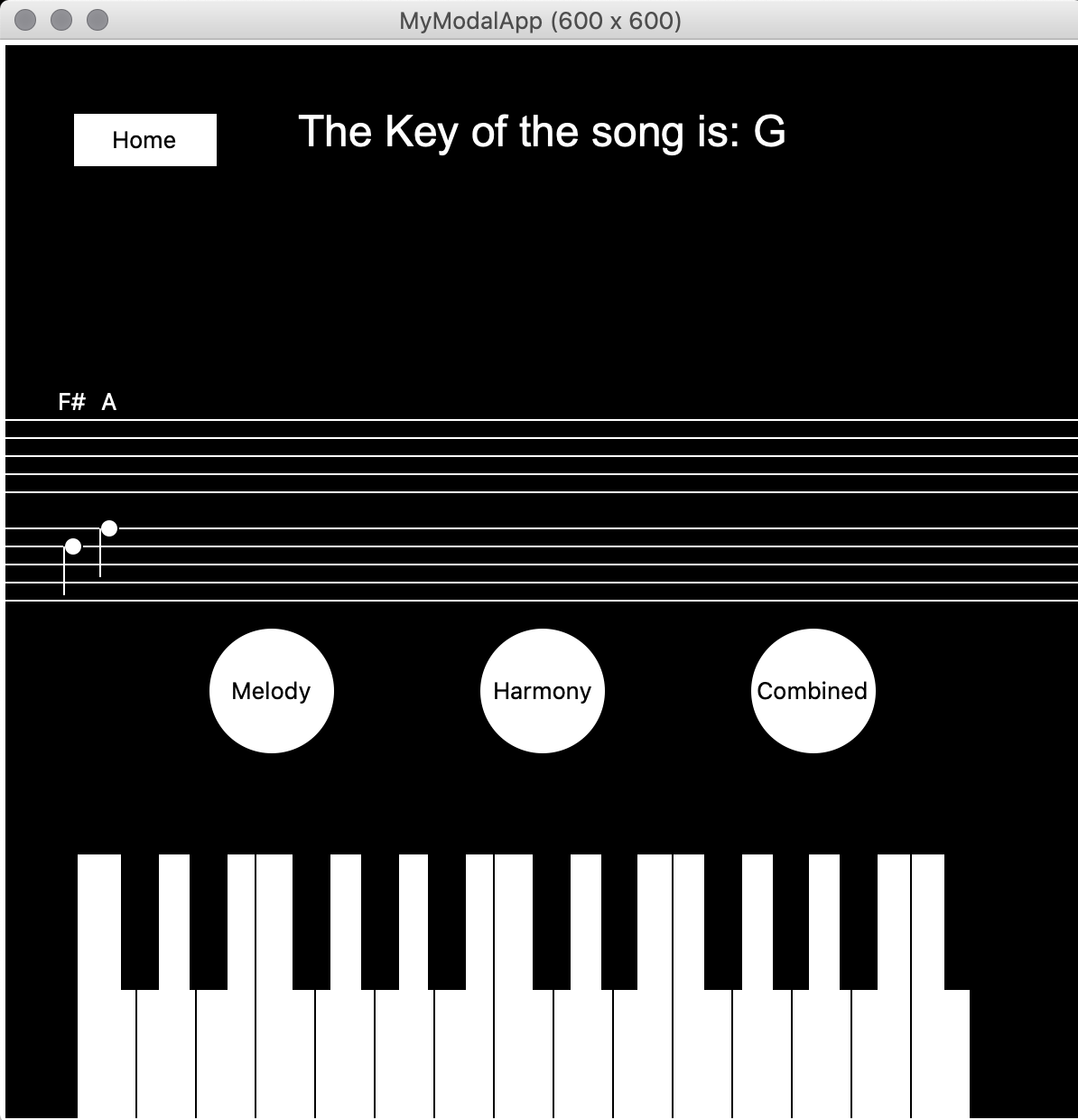
* Formatted the code for the checkpoint, worked on some more bugs- 4 hours

Tuesday, Wednesday, Thursday

* Spent 7-9 hours researching on the visualizer and integrating the code from online
* Inverted colors, added comments and cleaned up code for the deliverable

Current issues/ future goals:

* The piano color keys
* Visualizer
* The darken record button



Tp3 update

* New external modules namely pylab was used for the visualizer
* The visualizer uses a html page
* There is an info page added to the code which displays information about how the sheet music relates to the piano notes.
* The piano can be played using key presses and mousepressed and it light up displaying the note that is being played
* The sheet music now displays the harmony notes.

Time sheet

* I changed the design and added meaningful information sections for the project -2 hours
* Worked on rewriting the code related to audio for 7 hours
* Changed the way the piano is being played 3 hours. Encountered some issues and fixed bugs
* Noticed that there were some mistakes in the sheet music, added treble and bass clef
* Comments and final editing of code – 6 hours
* Tp video – 5 hours

Updated design