Project Proposal

Project Description

The term project is a Guitar Tab Generator, which creates a guitar tab (a “sheet music” for guitar) based on microphone input. The purpose of this is to eliminate the need to manually create guitar tabs by hand, where instead guitar can be played into the microphone to generate one.

Competitive Analysis

This project shares similarities with TransPyser, another 112 Term Project from a previous year that generates sheet music by processing a MIDI keyboard’s input into notes while reading pitch. This deals with a direct wire connection into one’s computer, and visualizes pitch and notes in real time while one is playing.

The Guitar Tab Generator shares similarities in the concept of generating a guitar tab and visualizing pitch and notes in real time while playing, but varies in the method of acquiring input. TransPyser takes direct input, while the Guitar Tab Generator has to dissect microphone input and read pitch, volume, and create a guitar tab based on tempo and when new notes are played. It also includes a standard guitar tuner to allow the user to tune their guitar to standard tuning before attempting to generate a tab. Guitar Tab Generator might require the user to play slower than usual to make processing audio easier, since it is not taking direct input into the computer.

Structural Plan

The project is organized in a Term Project folder, which consists of three primary .py files: Main.py, AudioProcessing.py and Testing.py. It also includes cmu_112_graphics to create the UI for the program. Main.py is the primary file, containing the import of AudioProcessing.py and other libraries/modules to create the visual for the user. The structure is broken down into screens, where there is a main introductory screen, a guitar tuner screen, a recording screen and a tab screen. The AudioProcessing.py file is where most of the backend processing occurs. I process the microphone input and gather pitch, volume, and time, and have created methods to break up the processing process so these methods can be used in the Main.py file. It will contain the bulk of the guitar audio processing, and uses PyAudio, Aubio, Sounddevice and other libraries.

Algorithmic Plan

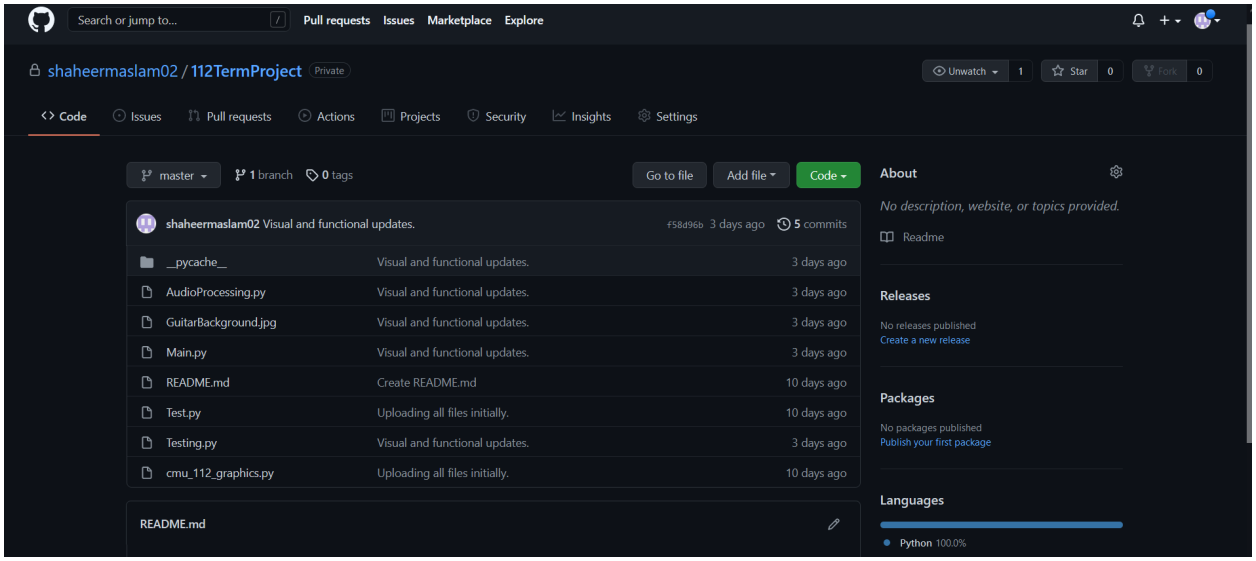
The trickiest part of the project is approaching how to measure pitch, tempo, volume and time to formulate a guitar tab from the variables I have. Since audio is measured from the microphone and not through direct input, we have more variables to consider when processing the audio and deriving a tab. Forming the guitar tab and functionality itself will require UI algorithmic complexity where one must be able to alter the tab itself and essentially be able to manipulate it however they please using arrow keys and commands along the tabs to place notes or not. I plan on developing my own algorithm to measure notes based on time passed, and when to process notes as “actual” notes based on changes in volume in the microphone.

Timeline Plan

TP1: I am hoping to have improved the UI and figured out an algorithm for creating guitar tabs by storing notes and measurements of volume and tempo.
TP2: I am hoping to have a working and functional version of the project where it creates guitar tabs decently well and the guitar tabs can also be manipulated by the user in case mistakes were made.
TP3: I will have finalized the project, improved the accuracy of creating guitar tab and fleshed out the UI experience.

Version Control Plan

I am using Github as my version control. I have let my Term Project folder be a Github repository, and am using the command line to commit and push changes.



Module List

I am using the following modules for my project:

- cmu_112_graphics
- tkinter
- sounddevice
- scipy
- aubio
- pyaudio
- numpy
- pandas
- requests

TP2 Update

No changes were made.

TP3 Update

- Created zero-crossing algorithm for pitch detection
- Created autocorrelation algorithm for cleaning and smoothing the audio samples by using a rolling average
- Created peak detection algorithm to find peaks, similar to idea of “onset” of a musical note, allowing me to match frequencies with peaks for a minimum threshold vol.
- Created simple guitar tab generation algorithm using the above three