Test Document

Table of Contents:

1. Introduction
    1.1. Purpose
    1.2. Scope
2. Specific Requirements Testing
    2.1 Data I/O
    2.2 Language Definition
    2.3 Time Level User Interface
    2.4 Score Level User Interface
    2.5 Conversion Tools
    2.6 Software restrictions
3. Stretch Requirements
    3.1 Grapheme to Phoneme
    3.2 Robust Linguistic Analysis

1.Introduction

1.1 Purpose

The purpose of the Singing Data Labeling Tool is to provide an easy and friendly interface for preparing singing data for machine learning. The goal of this document is to explain how we plan on testing each of the requirements that the client is expecting in detail.

1.2 Scope

In-Scope components:

a. Data Handling Features
b. Time Level User Interface Features
c. Score Level User Interface Features
d. Output Features
e. Automatic Labeling Features
f. Saving and Loading Features
g. Automatic Conversion Tools

These are the general features of the Singing Data Labeling tool that will be tested throughout development.

2. Specific Requirements Testing

2.1 Data I/O

2.1.1 Singing Data Format

A. Objective
   a. This test case verifies that the user can open an uncompressed WAV file and the program will read it
B. Inputs
   a. WAV file
C. Outcomes
   a. WAV should be displayed to user as visual audio signals

2.1.2 Music Score Import Formats

A. Objective
   a. This test case verifies that the user can import music scores from different files
B. Inputs
   a. ust, vsqx3, vsqx4, and musicxml files
C. Outcomes
   a. The music score should be displayed to the user, and be playable/modified

2.1.3 Output Format

A. Objective

a. This test case verifies that the program will allow output format to be user defined.

B. Inputs
    a. User selected output format options

C. Outcomes
    a. Output should be formatted with options--such as which label feature to iterate over and what to do on error, static text, and dynamic entries--that the user has chosen.

2.1.4 Project File

A. Objective
    a. This test case verifies that the project file contains the proper files associated to facilitate collaboration using version control.

B. Inputs
    a. Signal generated by user clicking button programmed for creation of projects

C. Outcomes
    a. This should create a project with paths to the audio, lebel data, output formats, and language definitions

2.2 Language Definition

2.2.1 Phoneme Definitions

A. Objective
    a. This test case verifies that the user will be able to define the sampa they are using as it relates to the IPA. Check requirements for definitions.

B. Inputs
    a. User's choices for defining the sampa

C. Outcomes
    a. Program should allow each user sampa to map the multiple IPA letters

2.2.2 Word Dictionary

A. Objective
    a. This test case verifies that user selected words can be converted to phonemes

B. Inputs
    a. Words user wants to convert into phonemes

C. Outcomes
    a. Program should use the implemented dictionary system to successfully convert the words to phonemes

2.2.3 Syllable Definition

A. Objective
    a. This test case verifies that the user can input a weighted table/graph to define syllables
B. Inputs
    a. Weighted table/graph provided by the user
C. Outcomes
    a. Program should analyze the table and model the phonotactic structure of the language.

2.3 Time Level User Interface

2.3.1 Waveform

A. Objective
    a. This test case verifies that the user will be able to zoom in and out, scroll, select regions, and playback regions
B. Inputs
    a. Signals generated by clicks made by the user
C. Outcomes
    a. Waveform should instantly reflect changes made by the user to the interface

2.3.2 Spectrogram

A. Objective
    a. This test case verifies that the spectrogram displays the in-depth information on the phonetic properties to the user.
B. Inputs
    a. Options the user has chosen for customization of color, contrast, hop size, and sample size
C. Outcomes
    a. Program should display the spectrogram with the modifications chosen by the user, as well as display the f0 to aid in identifying note transitions

2.3.3 Feature Layers

A. Objective
    a. This test case verifies that the layers are allowing the user to input time markers to input labels, and that the user adds additional layers from a list of preset types.
B. Inputs
    a. Time markers, and any additional layers the CL chooses from the preset list
C. Outcomes

a. The user inputted time markers should be appended to the input labels, with the same results on the added layers. The layers should also have a parent-child relationship to aid in grouping and access

2.3.4 Project Feature Layer

A. Objective
   a. This test case verifies that the program can overlay layers--one at a time--onto the waveform and spectrogram
B. Inputs
   a. Layers modified by the user
C. Outcomes
   a. The program should overlay the layers and allow the regions to cycle through colors to make them more distinct

2.3.5 Shortcuts

A. Objective
   a. This test case verifies that the program can implement the common tasks undo, redo, cut, copy, paste, and duplicate.
B. Inputs
   a. Key bindings or buttons on UI clicked
C. Outcomes
   a. The editor should implement the tasks correctly and properly update UI with changes instantly

2.4 Score Level User Interface

2.4.1 Piano Role

A. Objective
   a. This test case verifies that the provided piano tool has enough octaves to cover the theoretical range of human singing, and allow for notes to be snapped to a grid that can be modified for the note to snap to different sizes.
B. Inputs
   a. Notes that will be snapped onto the piano grid
C. Outcomes
   a. The piano role should instantly update the UI with the snapped notes and modifications made by the user. The grid should also have its rows color coded for sharps/flats and be offset to align to the audio.

2.4.2 Feature Overlays

A. Objective
   a. This test case verifies that the program will display an overlay of the waveform and f0 to aid in manual input
B. Inputs

a. Signal generated by user clicking the button to bring up the overlay

C. Outcomes

a. The program should display the waveform and f0

2.4.3 Notes

A. Objective

a. This test case verifies that the notes, which contain words and phonemes, can be resized and cut. Also verifies that phonemes can be locked separate to words, and each note can access additional features

B. Inputs

a. Notes that the user will input for editing

C. Outcomes

a. The program should recognize the actions the user wants to perform on the notes and instantly update the UI to reflect them. Notes should also have the additional features applied to them such as tempo.

2.4.4 Playback

A. Objective

a. This test case verifies that the playback feature of the programs works on selected regions and tones for notes to be used as a reference

B. Inputs

a. Select region or note to be played

C. Outcomes

a. The program should playback the audio of the region selected and play the tone of the specified note.

2.5 Conversion Tools

2.5.1 Lyrics to Phonemes

A. Objective

a. This test case verifies that the program converts lyrics to phonemes

B. Inputs

a. Lyrics provided by the user

C. Outcomes

a. The program should use the provided dictionary data to convert lyrics into phonemes to be aligned

2.5.2 Phonemes to Syllables to Notes

A. Objective

a. This test case verifies that the program can output a list of syllables and assign each a note using the provided graph

B. Inputs
    a. Graph with data pertaining to phonemes
C. Outcomes
    a. The program should estimate note identity based on average pitch at that moment and assign a note to each syllable.

2.6 Software restrictions

3.6.1 Portability

A. Objective
    a. This test case verifies that the program can run on Windows, Linux, and Mac OS
B. Inputs
    a. N/A, we will run the program on all 3 platforms for testing
C. Outcomes
    a. Program should act the same no matter the OS

3. Stretch Requirements

3.1 Grapheme to Phoneme

A. Objective
    a. This test case verifies that the program can take in written words and store after being ran through an algorithm output predicted phonemes
B. Inputs
    a. Written word provided by the user
C. Outcomes
    a. Program should output predicted phonemes of the user's input

3.2 Robust Linguistic Analysis

A. Objective
    a. This test case verifies that the program can generate syllables using a more featured tool set such as a decision tree and rules based on phoneme properties
B. Inputs
    a. Phonemes provided by the user
C. Outcomes
    a. The program should generate syllables based on the phonemes provided