Requirements Document
EECS2311 Software Development Project

Group 14

Adil Hashmi (adilhashmi.w@gmail.com)
Alexander Arnold (alex290@my.yorku.ca)
Boho Kim (kimboho614@gmail.com)
Kanwarjot Bharaj (kjsingh76@gmail.com)
Stanley Ihesiulo (stanihe1901@gmail.com)

Table of Contents

[Introduction]

This document will give an overview of all the list of requirements needed to run the MusicXML application successfully. The application is built and designed using Java, Fxml, and Gradle. Compatible with both Eclipse and IntelliJ. The software includes features such as a graphical user interface (GUI), easy to use with an intuitive design. A MusicXML file generator, and finally an end-to-end object-oriented translating algorithm. The algorithm translates tablatures which are user inputted or entire .txt files. This software is designed for three stages with the first one being to get user input, then translating input and generating MusicXML, and finally creating a new file with the final product ready for use.

[Product Scope]

The text Tablature is in the form of musical notation indicating instrument fingering rather than musical pitches (https://en.wikipedia.org/wiki/Tablature). It is easy to read and practice, but it is incompatible with music applications such as MuseScore; unable to be converted to sound information. Therefore, our project is converting the text file to MusicXML file format which can be converted by music the application. The user can get sound information from the Tablature text file using music the application.

[Use Cases]

1. Converting a Guitar Text Tab

Primary Actor: User
Goal: User wants to convert text tab into MusicXML file
Success Scenario:

1. User starts application
2. User inputs the .txt file into system.

3. System identifies type of instrument
4. User tells system to convert text tab by clicking ‘convert’ button
5. System converts the .txt file into a MusicXML file
6. System creates a new output file
7. User locates file
8. User closes application

Precondition: User already has a guitar tablature saved onto their computer

2. Personal preference of score type
Primary Actor: User
Goal: User wants to choose score type to convert
Success Scenario:

1. User starts application
2. User inputs the .txt file into system
3. User chooses score type to convert
4. User tells system to convert text tab by clicking ‘convert’ button
5. System converts the .txt file into a MusicXML file with type which user wants
6. System creates a new output file
7. User locates file
8. User closes application

Precondition: User already has tablature saved onto their computer

3. Fixing tablature to appropriate format and converting it.
Primary Actor: User
Goal: User wants to see if text format which user wants to convert is appropriate
Success Scenario:

1. User starts application
2. User inputs the .txt file into area provided by copying and pasting
3. System recognizes if each text line is proper to convert or not
4. If there are unrecognizable symbols and lines, highlight it and let user know problems when the mouse pointer hovers over it
5. User fix or remove unrecognizable symbol from text
6. User tells system to convert text tab by clicking ‘convert’ button
7. System creates a new output file
8. User locates file
9. User closes application

Precondition: User already has tablature saved onto their computer

[User stories]

- As a user, I want to convert tablature text file to MusicXML file to run it in musical application.

- As a user, I want to convert tablature text file to MusicXML file with specific score type that I want to see in musical application.

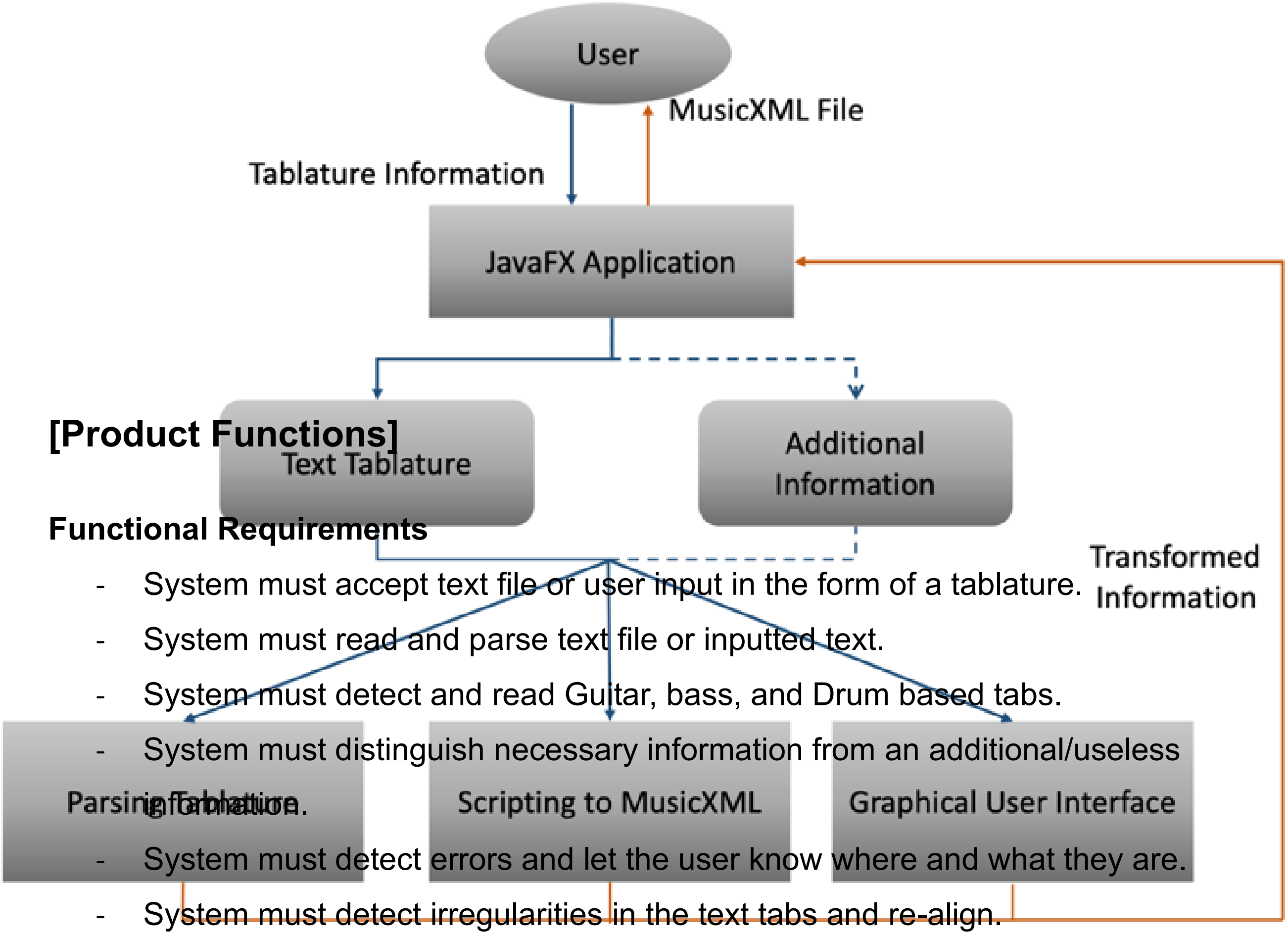
- As a user, I want to see which symbols and lines is not appropriate to convert in tablature text file, so that I can fix it using system and convert it.

[Overall description]

We receive information from user by using JavaFX application. The mandatory information is Text Tablature (File or typing). Additional information could be asked such as song name, beat, beat-type, kind of instrument and so on if we inevitably need it to transform information (But, we give priority to not asking as much as possible). After transforming information, we return output as MusicXML file format to user.

[User Characteristics]

Even though the Tablature form has not been clearly organized by people, the user should provide with text information based on general Tablature form. For the case of asking additional information about music material such as beat, beat-type or so on, the user should have basic musical knowledge.



[Product Functions]

Functional Requirements

- System must accept text file or user input in the form of a tablature.
- System must read and parse text file or inputted text.
- System must detect and read Guitar, bass, and Drum based tabs.
- System must distinguish necessary information from an additional/useless information.
- System must detect errors and let the user know where and what they are.
- System must detect irregularities in the text tabs and re-align.

- System must take into account drop tunings.
- System must take into account different time signatures.
- System must prompt user for necessary or creative input if and when needed.
- System must take into account hammer-ons, bends, pull-offs, and slides.
- System must build a MusicXML file given the tablature.

Constraints

- Limited information (To minimize asking users additional information).
- Various form of Tablature.
- Knowledge about music.
- REGEX.

Non-Functional Requirement

Usability/Performance

- System must have an intuitive GUI (User friendly).
- System must have a fast conversion response time.

Quality/Reliability

- System must produce an accurate (Error free) MusicXML file.
- System must be compatible with various types of tablatures.
- System must be compatible with two operating systems.

Supportability

- System must allow user to have control of how they want their tablature to be built (Preferences, time signatures).