Software Requirements Specification

for

Music Shuffle

Version 1.0 approved

Prepared by Lucky Abolorunke, Indigo Jones, and Charles Owens

Group 2

04/05/2021

Table of Contents

Table of Contents ii
1. Introduction 1
1.1 Purpose 1
1.2 Document Conventions 1
1.3 Intended Audience and Reading Suggestions 1
1.4 Product Scope 1
1.5 References ..... **Error! Bookmark not defined.**
2. Overall Description 2
2.1 Product Perspective 2
2.2 Product Functions 2
2.3 User Classes and Characteristics 2
2.4 Operating Environment 2
2.5 Design and Implementation Constraints 2
2.6 User Documentation 2
3. System Features 3
4.1 System Features 1 3
4.2 System Feature 2 (and so on) 3

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

This following document is in reference to the Music Shuffle which is the product that is made and described. The Music Shuffle is only a part of the system. Furthermore, it is a small java program.

1.2 Document Conventions

*This system will involve three main classes that are between high and medium priority.The **Shuffle class** which has higher-level needs of requirements due to the fact that it will be executing the main task of the program. As for the **MusicShuffle class**, it has a medium priority due to the fact that it is the simple display of results after the **Shuffle class** is called to action. In addition, we have the **Button class,** which sets off and displays the desired output that the user is expecting one the button is clicked on.And finally you have the **MusicShuffleTester class** also has medium priority for its task is to check and test out the two classes. As for variables and functions names, they are not currently specified in this document.*

1.3 Intended Audience and Reading Suggestions

The Music shuffle covers a wide variety range in regards to the audience: marketing staff, developers, and testers. But the main target audience is the users. This SRS contains the inner and outer workings of the program. It is organized by the following in the following order: the functionality of the program, the specific implications and components embedded into the program, and an overview of instructions and directions for the user to follow and understand the limitations of this program.It is suggested that you read from sections 1.3-3.1 in chronological order to grasp full understanding of the product.

1.4 Product Scope

The Music Shuffle is a program where the user is able to receive a shuffled playlist to play for a duration of time provided by the user. This shuffle playlist will not include any repeating songs and will allow the user to enjoy a unique playlist every time they shuffle it. With this software, it will give my team the ability to provide a more convenient way of enjoying their experience with music.

Date: Time	*Task*	*Team Member Responsible*
Start: 4/11/2021	***Shuffle class*** ***Button Class***	*Lucky*
End: 4/28/2021		
Start:4/27/2021	***Music Shuffle class*** *(output gui)*	*Charles*
End: 5/04/2021		
Start: 4/11/2021	*tester classes (**MusicshuffleTester** class)*	*Indigo*

End 5/04/2021		

2. Overall Description

2.1 Product Perspective

The Music Shuffle is a replacement for an existing system that is popular in many music-playing services and devices. While it is usually a feature of a larger system for playing music, in this case, it is it's own product with a music shuffler being its only feature. Its major components are taking playlists and time durations for inputs, shuffling that playlist for the given amount of time, and outputs the shuffled playlists.

2.2 Product Functions

- *Input field that lets the user input an amount of time they want to listen to their playlist*
- *A playlist of songs that the user can select to shuffle through*
- *A "shuffle" button for user to randomize the songs in the playlist*
- *An uneditable playlist of the randomized order of the user's selection of songs*

2.3 User Classes and Characteristics

Since this product is generally more simplistic than most, users will mostly have the same characteristics. Young users will need the functions to be fast so that it will fit the same standard as most music programs today. Middle aged - older users will most likely need the functions to be clear and obvious as to what each function's use is. The target user base is teenagers and young adults. Developers will have full access to the software code.

2.4 Operating Environment

The software will operate on the Visual Studio Code environment but will be available to use on MacOS and Windows for users. Any modern computer will be able to run the software.

2.5 Design and Implementation Constraints

While most modern computers will be able to access this shuffle, older computers will have trouble handling this software. This software will also not be available on any software outside of MacOS or Windows (No Linux). The music shuffle also is only in English, so non-English speakers will have trouble trying to use this software. The developers of the shuffle are responsible for maintaining the delivered software. The application was created with JFrame, Visual Studio Code, Replit, and edited across Windows, Chrome OS, and MacOS. Developers not on the original team must ask permission for any type of editing or viewing of our application's code. New developers must also have a working knowledge of Java, as it was the only programming language used for the application.

2.6 User Documentation

The instructions next to the shuffle button are the only instructions given for this application. If the user needs additional help, they can contact one of our developers.

3. System Features

The system has a user interface that gives the user an option to select from a list of songs on their playlist that they want to shuffle; Also, the user gets a space to put in the length in minutes of the music you want to listen to; and then a shuffle button.

3.1 System Feature

The List of Music

- *Description and Priority:*
  The GUI displaces a list of 20 songs in which the user can select from. The user has the options of picking 0 or 20 songs.
- *Functional Requirements:*
  The songs are displaced on the GUI through the JList class. The user has the options to select the multiple songs, and unselect songs.

The Text Field

- *Description and Priority:*
  The Text Field allows the user to input a number, the time length of the song they want to listen to too. This number can range for zero to infinity.
- *Functional Requirements:*
  The function has a JTextField declared, to display a box for the user to input a number.

The List of Music

- *Description and Priority:*
  The GUI displaces a list of 20 songs in which the user can select from. The user has the options of picking 0 or 20 songs.
- *Functional Requirements:*
  The songs are displaced on the GUI through the JList class. The user has the options to select the multiple songs, and unselect songs.

The Button

- *Description and Priority:*
  The button class is used to shuffle the songs selected by the songs.
- *Functional Requirements:*
  The Button function implements the shuffle class whenever it has been clicked. It has an action click listener that executes the shuffle class

The Resulting Songs

- *Description and Priority:*
  The once the user clicked the button the JList field is created to displace the songs