|  |
| --- |
| **DOCUMENTATION** |
| **JMC MUSIC PLAYER** |
| Java III |

|  |
| --- |
| Jose Rico Imbang  12-3-2020 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version #** | **Implemented By** | **Revision Date** | **Approved By** | **Approval Date** | **Reason** |
| 1.0 | Jose Rico Imbang | 02/12/2020 | Ken Beck | 02/12/2020 |  |

# TABLE OF CONTENTS

[1 INTRODUCTION 1](#_Toc57903634)

[1.1 Purpose of the Product Design Specification Document 1](#_Toc57903635)

[2 GENERAL OVERVIEW AND DESIGN GUIDELINES/APPROACH 2](#_Toc57903636)

[2.1 Assumptions/Constraints/Standards 2](#_Toc57903637)

[3 ARCHITECTURE DESIGN 3](#_Toc57903638)

[3.1 Logical View 3](#_Toc57903639)

[3.2 Hardware Architecture 3](#_Toc57903640)

[3.3 Software Architecture 3](#_Toc57903641)

[3.4 Security Architecture 4](#_Toc57903642)

[3.5 Performance 4](#_Toc57903643)

[4 SYSTEM DESIGN 5](#_Toc57903644)

[4.1 Use-Case 5](#_Toc57903645)

[4.2 Database Design 5](#_Toc57903646)

[4.3 User Interface Design 6](#_Toc57903647)

[5 TEST DATA AND EVIDENCE 7](#_Toc57903648)

[5.1 Unit Testing with Junit 7](#_Toc57903649)

[5.2 Test Table 8](#_Toc57903650)

[6 SOURCE CONTROL 15](#_Toc57903651)

[6.1 GitHub 15](#_Toc57903652)

## INTRODUCTION

### 1.1 Purpose of the Product Design Specification Document

The Product Design Specification document tracks the necessary information required to effectively define architecture and system design in order to give the development team guidance on architecture of the system to be developed. The Product Design Specification document is created during the Planning Phase of the project. Its intended audience is the project manager, project team, and development team. Some portions of this document such as the user interface (UI) may on occasion be shared with the client/user, and other stakeholder whose input/approval into the UI is needed.

## GENERAL OVERVIEW AND DESIGN GUIDELINES/APPROACH

### 2.1 Assumptions/Constraints/Standards

The application will be tested using the Client’s computer (my computer) which runs Windows 10 Operating System. It has programs installed that will be used testing the project like JDK, JavaFX, and NetBeans. A web browser will also be used to open the help file.

[Java Code Conventions](https://www.oracle.com/technetwork/java/codeconventions-150003.pdf) will be utilised for the project to improve readability of the software, allowing engineers to understand the code more quickly and thoroughly. Code conventions helps in maintaining applications since hardly any software is maintained for its whole life by the original author, plus the fact that 80 % of the lifetime cost of a piece of software goes to maintenance. Code conventions also ensure that the source code is well packaged and clean.

The songs that will be played are in WAV format and other audio formats will not be tested.

## ARCHITECTURE DESIGN

### Logical View

The application will be installed into the computers that the user wishes to use it.

### 3.2 Hardware Architecture

The computers’ minimum hardware specification that is required to run the application are the following:

|  |  |
| --- | --- |
| Processor | 1 GHz |
| Memory | 2 GB |
| Storage | 32 GB |
| Graphics card | DirectX 9 or later with WDDM 1.0 driver |
| Display | 800x600 |

The client’s computer exceeds the minimum hardware specification that is required.

### 3.3 Software Architecture

The paths of the songs are saved in a CSV file. Once the Add Songs button is clicked, the information from the CSV file will be fetched sing a 3rd party library and each path will be saved into a Node of a Binary Tree data structure. The paths will also be shown in the Table View. The user then can search for a song using a Text Field. The user can click the Search button once the path of the song is entered into the Text Field. The song will be played if found and a Text Field on the upper part of the scene will flash the path of the song, otherwise, a friendly message will show up. The user can also sort the songs alphabetically by clicking the Sort button. Clicking the Save button will save the information into another CSV file. A help file can be accessed through a menu bar.

### 3.4 Security Architecture

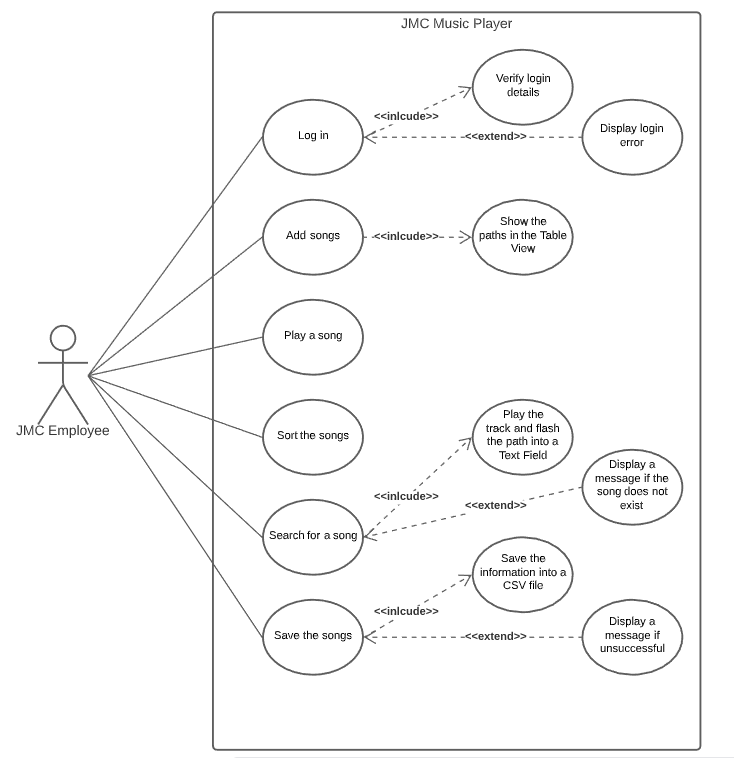
To be able to use the JMC Music Player application, the user needs to login with a username and a password. The username is in plain text while the password is hidden characters. The username is verified as it is while the password is hashed before it is verified and compared to the default Admin credentials.

### 3.5 Performance

A CSV file contains all the songs’ information and a 3rd party library will be used to read this information from the CSV file. Opening the application will store the songs into a binary tree and the GUI will show the track list. The user can search a song and the search algorithm utilised is binary search. The songs can also be sorted using merge sort algorithm.

## SYSTEM DESIGN

### Use-Case

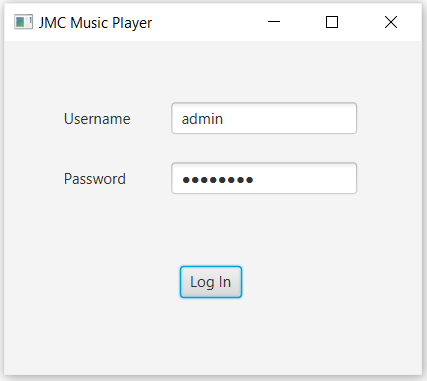


### Database Design

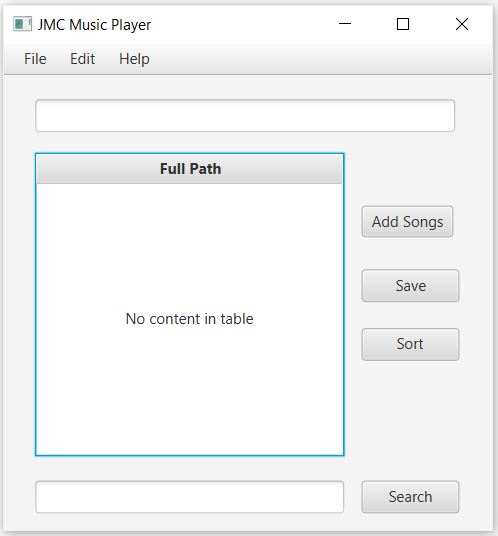
A CSV file will contain the paths of the songs.

### User Interface Design

Log In Interface

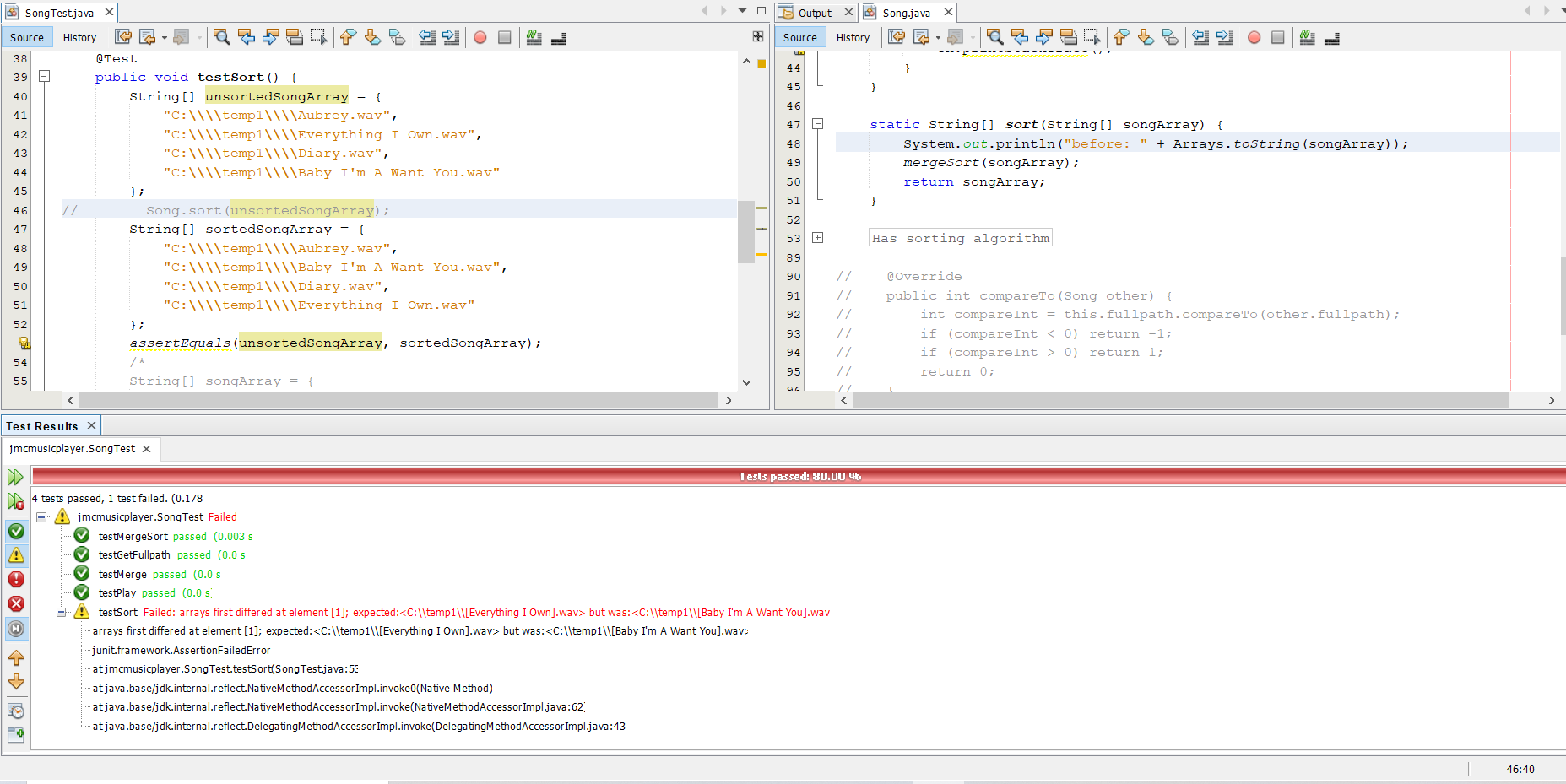


Music Player Interface

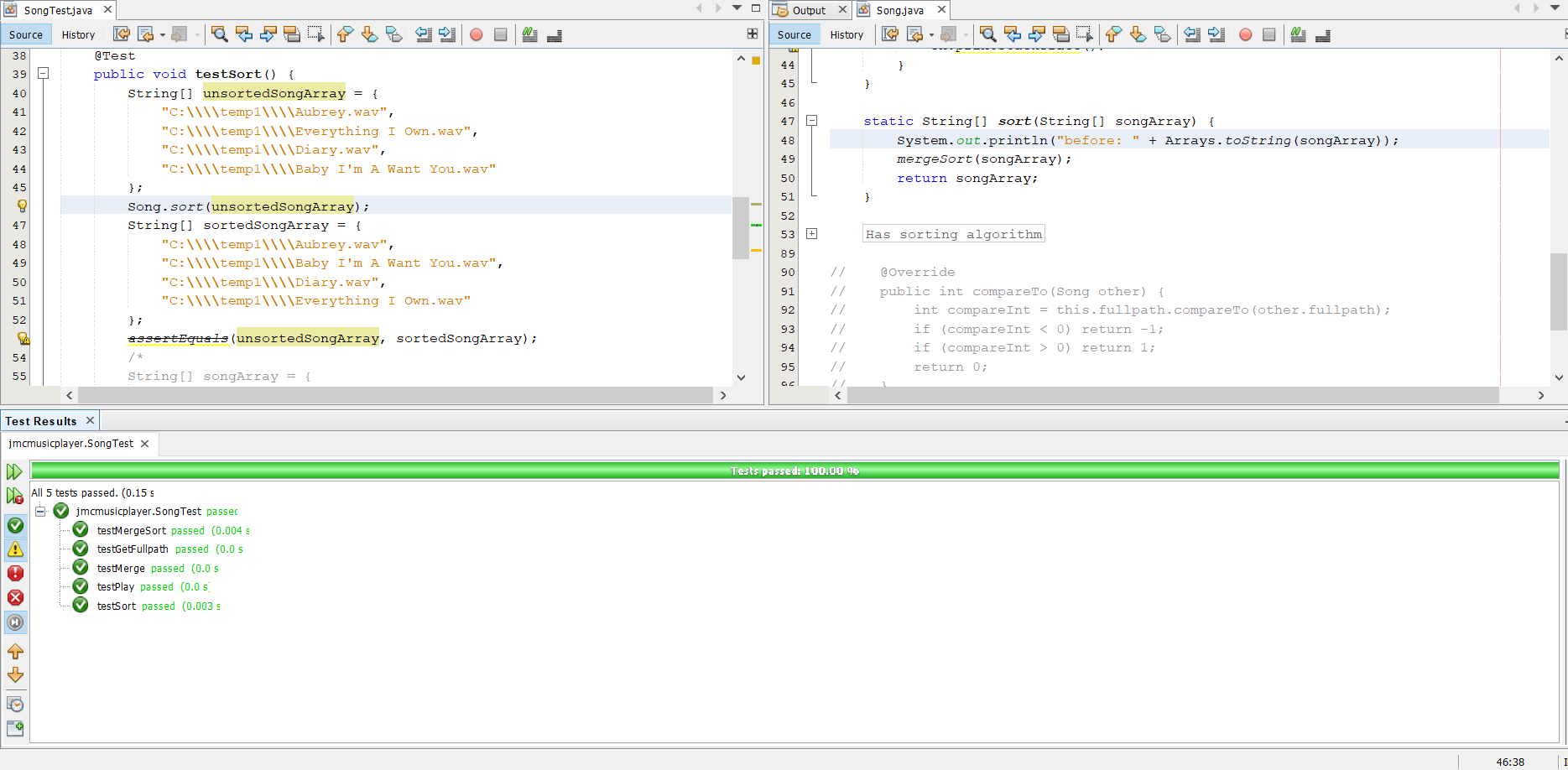


## TEST DATA AND EVIDENCE

### Unit Testing with Junit



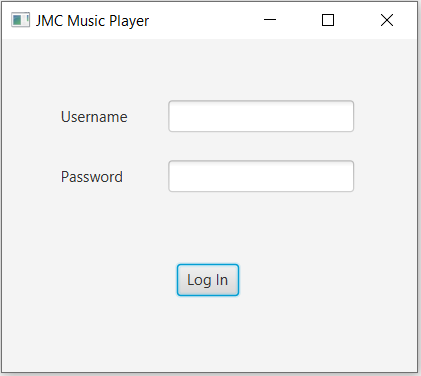
Got a 1 test failed result BEFORE using the *sort* method



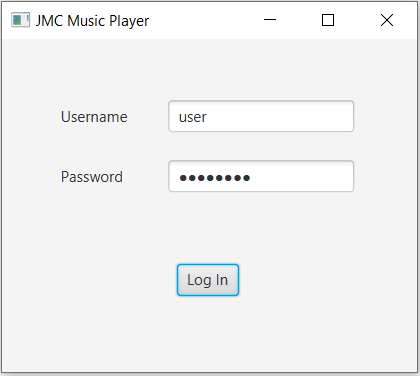
No failed test after using the *sort* method

### Test Table

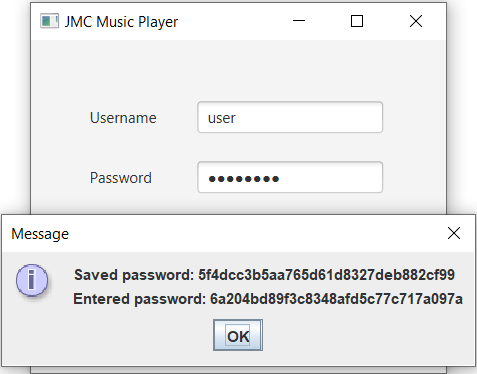
|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **Description** | **Expected Outcome** | **Evidence** |
| Case 1 | Run the application | The user is asked to login with the username and the password. | Ref 1 |
| Case 2 | Enter incorrect login information | A message shows up showing the hashed saved and entered password FOR DEMO.  Another message is shown saying the login information is incorrect. | Ref 2, 3 & 4 |
| Case 3 | Enter correct login information | A message shows up showing the hashed saved and entered password FOR DEMO then the Music Player is shown. | Ref 5 & 6 |
| Case 4 | Click Add Songs button | The information for songs.csv are loaded containing the unsorted paths of the tracks and are shown in the table. | Ref 7 |
| Case 5 | Type an incorrect path of a song into the Text Field and click Search button | A friendly message is shown. | Ref 8 |
| Case 6 | Type the correct path of the song into the Text Field and click Search button | The song is played, and its full path is shown in the upper Text Field. | Ref 9 |
| Case 7 | Click the Sort button | The paths in the table are sorted alphabetically. | Ref 10 |
| Case 8 | Click the Save button | A message pops up saying whether it is successful or not. If successful, the information in the table are saved into a CSV file named songs copy. | Ref 11 & 12 |
| Case 9 | Click Help from the menu bar then click About | A help file is shown. | Ref 13 |



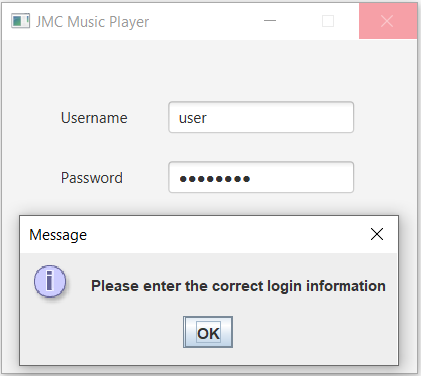
Ref 1



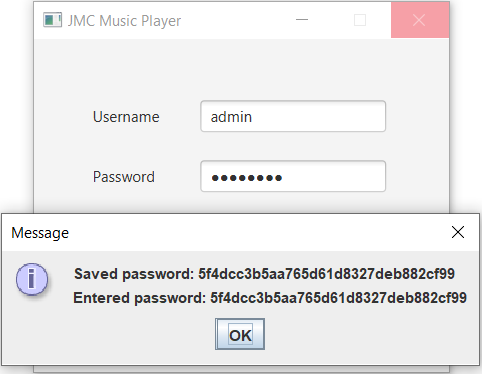
Ref 2



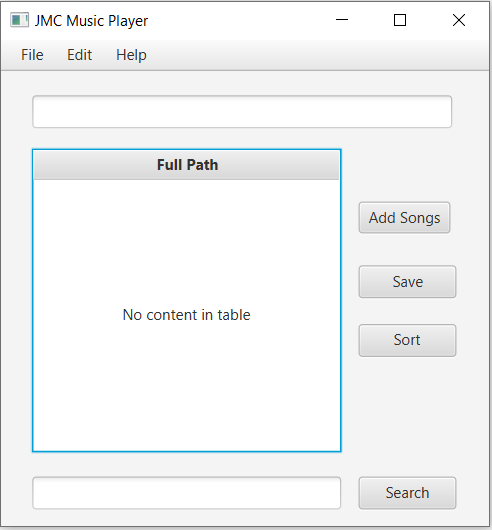
Ref 3



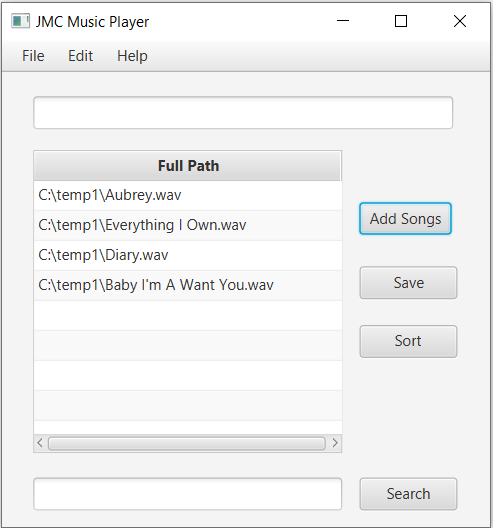
Ref 4



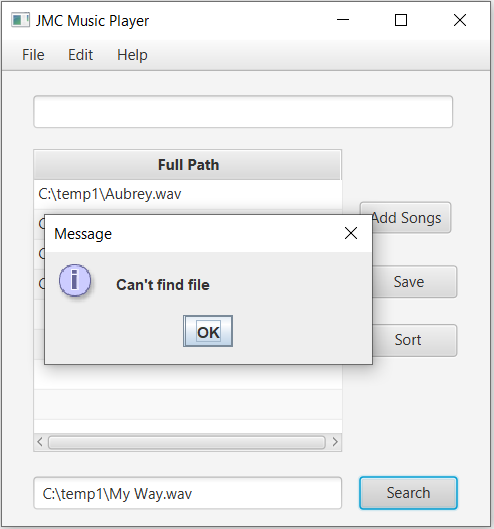
Ref 5



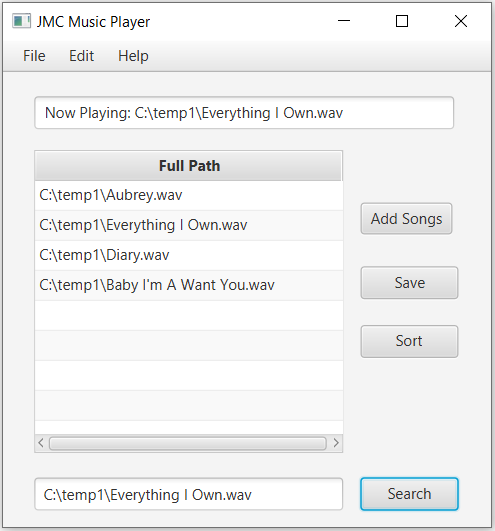
Ref 6



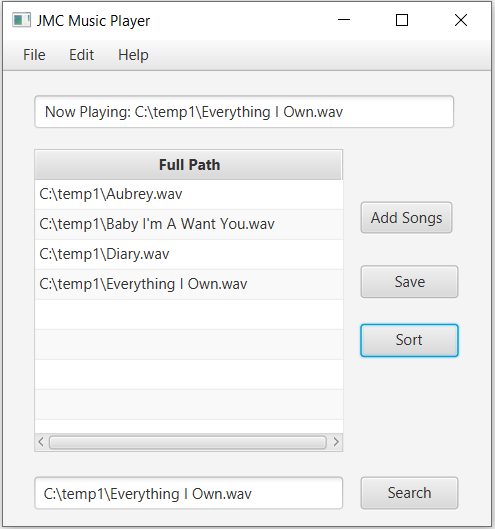
Ref 7



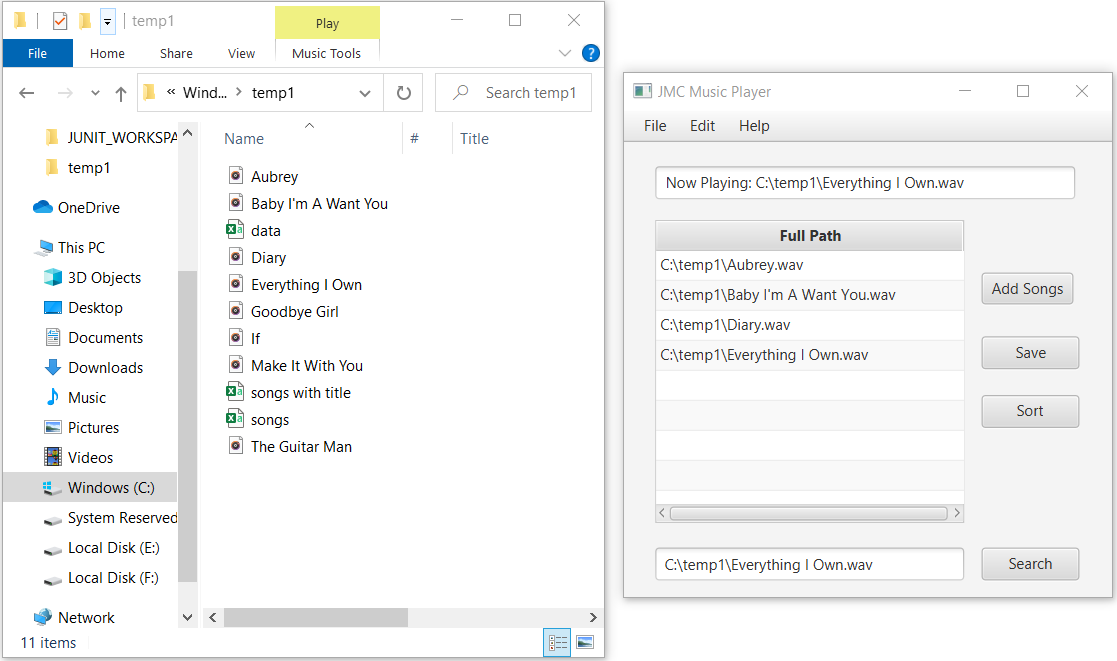
Ref 8



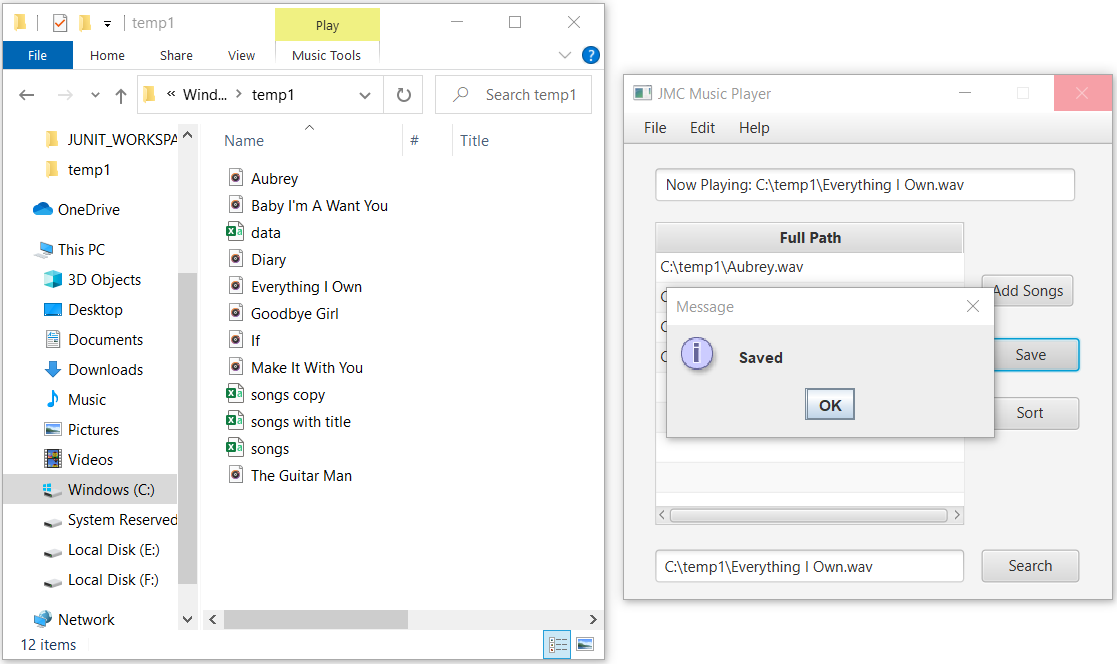
Ref 9



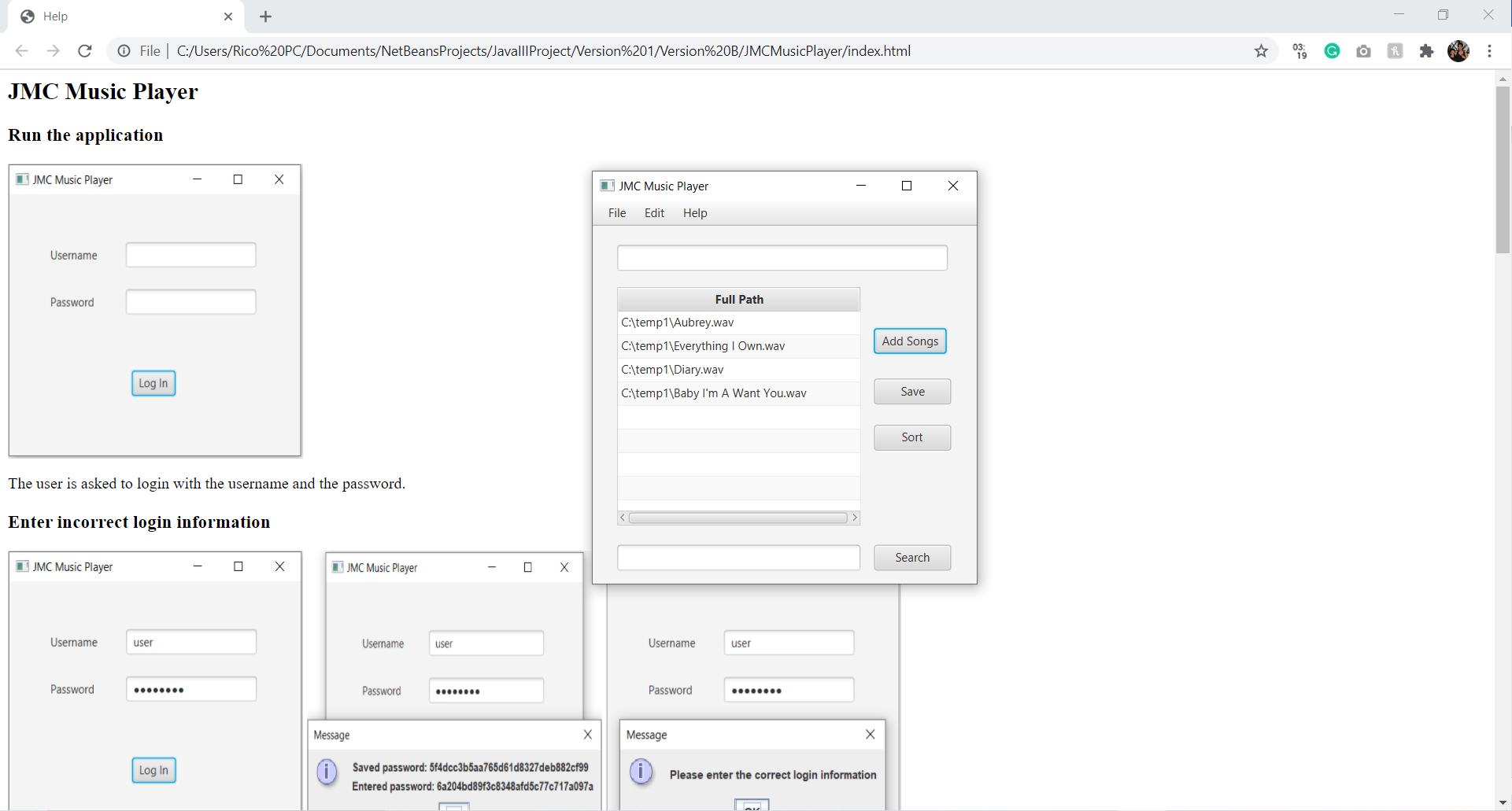
Ref 10



Ref 11



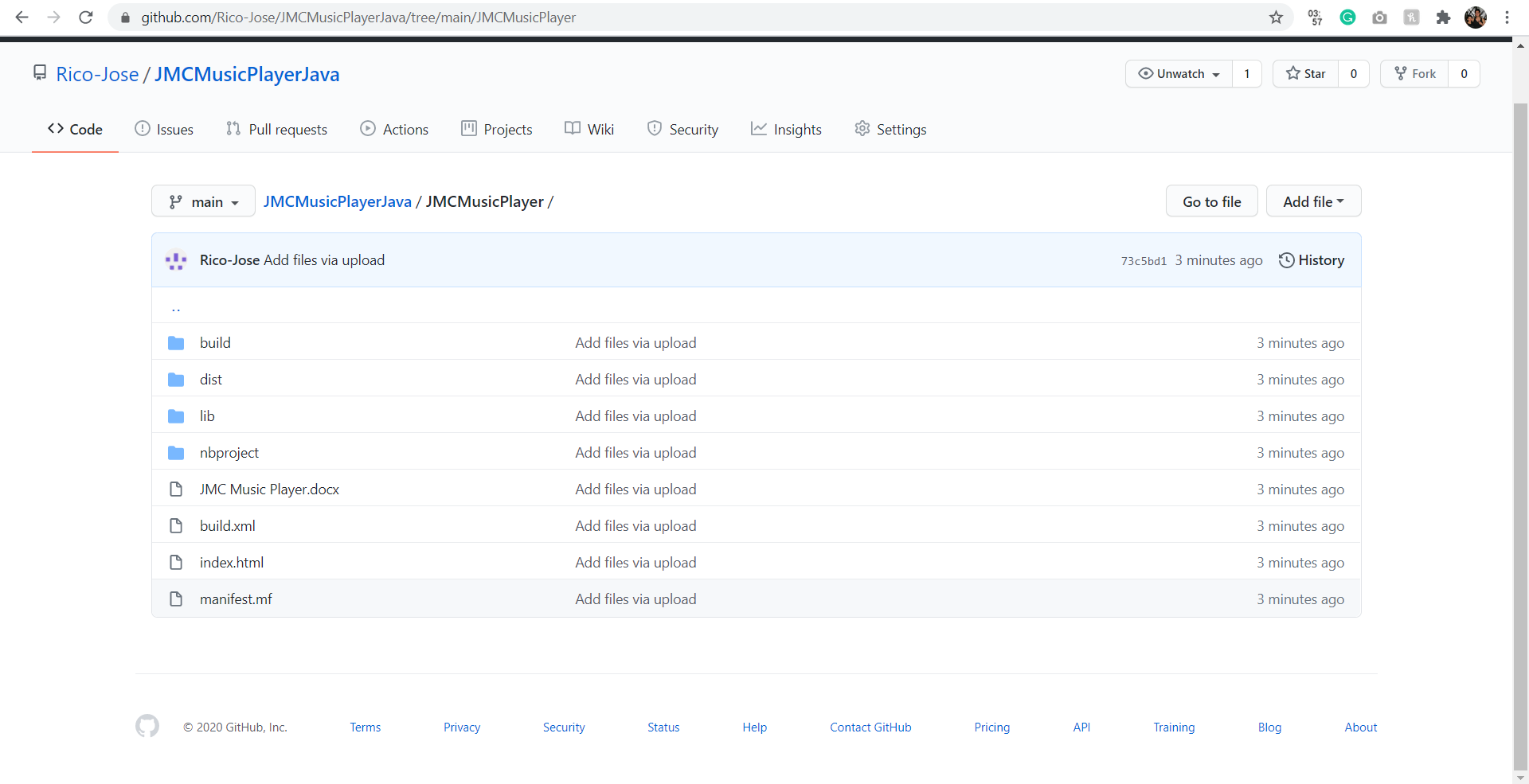
Ref 12



Ref 13

## SOURCE CONTROL

### GitHub



[Please use this link to go to the Repository](https://github.com/Rico-Jose/JMCMusicPlayerJava/tree/main/JMCMusicPlayer)