AT3 – Project  
JMC Music Player

Product Design Specification

Student:

Reece Pieri

Student ID:

M087496

Course:

Diploma of Software Development – Java III

Contents

[1 Introduction 3](#_Toc57241858)

[1.1 Purpose of the Product Design Specification Document 3](#_Toc57241859)

[2 General Overview and Design Guidelines/Approach 4](#_Toc57241860)

[2.1 Assumptions / Constraints / Standards 4](#_Toc57241861)

[2.2 Quality Assurance Practices 4](#_Toc57241862)

[2.3 Testing 4](#_Toc57241863)

[3 Requirements 5](#_Toc57241864)

[4 Architecture Design 6](#_Toc57241865)

[4.1 Logical View 6](#_Toc57241866)

[4.2 Hardware Architecture 9](#_Toc57241867)

[4.3 Software Architecture 9](#_Toc57241868)

[4.4 Security Architecture 10](#_Toc57241869)

[4.5 Communication Architecture 10](#_Toc57241870)

[4.6 Performance 10](#_Toc57241871)

[5 System Design 11](#_Toc57241872)

[5.1 Use-cases 11](#_Toc57241873)

[5.2 User Interface Design 11](#_Toc57241874)

[5.3 System Requirements 13](#_Toc57241875)

[6 Implementation Plan 14](#_Toc57241876)

[6.1 Goal 14](#_Toc57241877)

[6.2 Tasks 14](#_Toc57241878)

[6.3 Resources 14](#_Toc57241879)

[6.4 Responsibilities 14](#_Toc57241880)

[6.5 Success Metrics 14](#_Toc57241881)

[7 Source Control 15](#_Toc57241882)

[8 Test Plan 16](#_Toc57241883)

[9 Product Design Specification Approval 17](#_Toc57241884)

# 1 Introduction

* 1. Purpose of the Product Design Specification Document

The purpose of the Product Design Specification document is to store detailed information regarding the application design and architecture. It is to be used by the development team to provide guidance on how to build the application to meet the software requirements.

# 2 General Overview and Design Guidelines/Approach

2.1 Assumptions / Constraints / Standards

The application must be designed to:

1. Be responsive; the application must run smoothly. The GUI must not lock up or freeze during loading.
2. Use minimal resources; the application must use as little system resources as possible and be able to run in the background for long periods of time.
3. Look sharp; the design of the application must have a sharp, striking appearance. The use of standout colours is accepted and system controls appearances must be avoided.

2.2 Quality Assurance Practices

Software verification will be used during development to ensure the application is being developed in line with the client’s requirements.

Software validation will be used at the end of development to ensure that the final prototype will satisfy all of the requirements outlines by the client and that the client will be satisfied with the end product.

2.3 Testing

White box testing will be performed as features are added into the application to ensure that they are meeting the client’s requriements and that development is on the right track.

Black box testing will be completed in order to ensure that all of the client’s requirements are met and that the client will be satisfied.

# 3 Requirements

Dynamic Data Structure

A linked list will be used to store Song objects in a user’s playlists.

Hashing Technique

SHA-512 hashing will be used to store user passwords in User objects on the server.

Sorting Algorithm

Merge Sort will be used to sort songs in a playlist.

Search Technique

Binary Search will be used to search for songs in a playlist.

Third-party Library

OpenCSV: Used to save a playlist in .csv format. Documentation and download can be found here at <https://www.baeldung.com/opencsv>.

Source Control

GitHub will be used for the source control for this project.

Coding Standards

Traditional coding standards apply to this application to ensure maximum code readability for easy maintenance and re-use. This includes following naming conventions, indentation, commenting, code grouping and more.

Testing

White box testing will be performed as features are added into the application to ensure that they are meeting the client’s requriements and that development is on the right track.

Black box testing will be completed in order to ensure that all of the client’s requirements are met and that the client will be satisfied.

# 4 Architecture Design

The server and client applications will interact during login to verify user credentials and allow the user access to the music player. Connection between the client and server will be achieved via sockets.

During playlist load and save, the client will interact with the system file explorer through Open and Save dialog windows.

4.1 Logical View

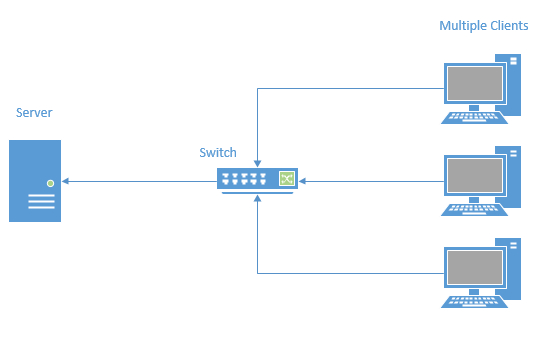


Figure 1 – Hardware architecture

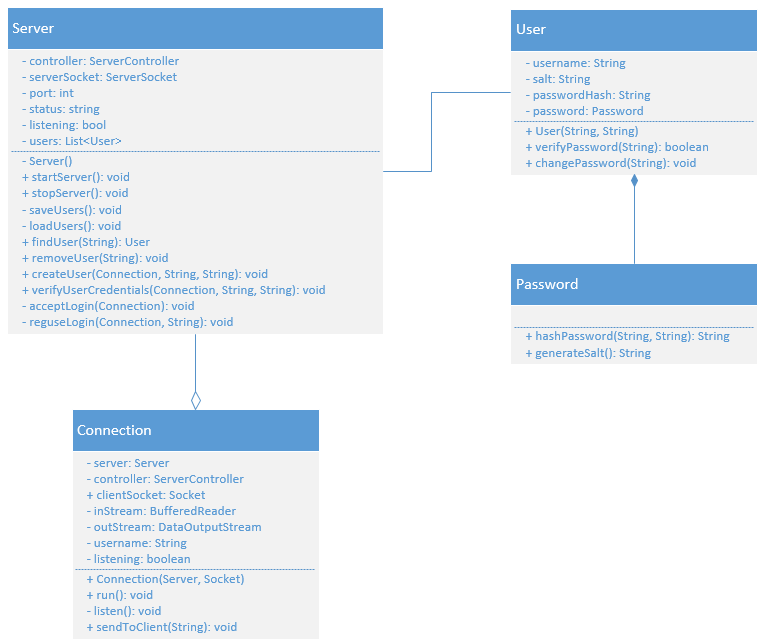


Figure 2 – Software architecture (Server)

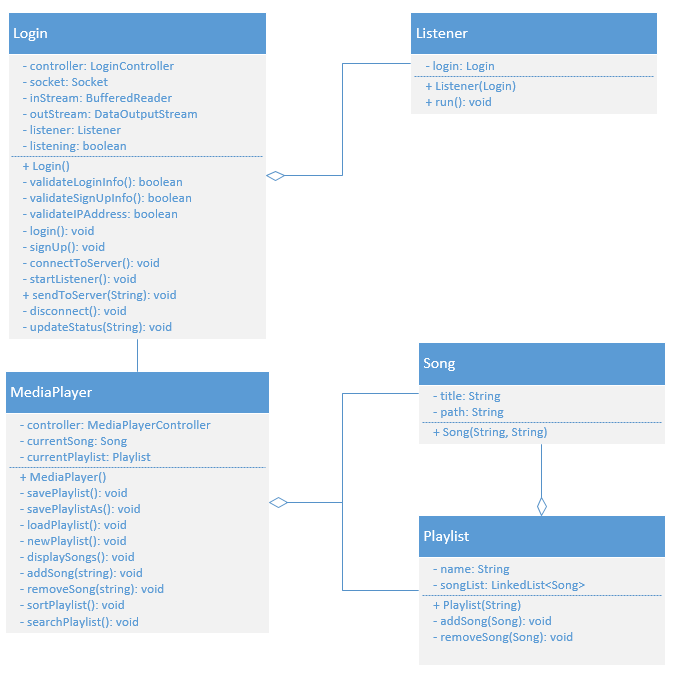


Figure 3 – Software architecture (Client)

4.2 Hardware Architecture

The Server application will run on a virtual machine on a central server.

The Client application will run on PC’s and connect to the server via a switch on their local network.

Refer to Figure 1.

4.3 Software Architecture

This desktop application is designed to run on operating systems capable of running Java applications.

Server

The server application will create User objects which will be loaded/saved on server start/stop respectively. User objects will be stored in a binary file.

The server will also verify user login credentials on client login attempts.

Refer to Figure 2.

Client

Clients will be able to login and create an account through the Login GUI. On successful login, the MediaPlayer GUI will be displayed.

In the MediaPlayer GUI, users will be able to add songs to a playlist, which are stored in a LinkedList<Song>. The playlist is displayed in a listview.

Songs searches are actioned using a binary search.

Playlist sorting is achieved through a merge sort.

Playlists are saved to a .csv file.

Refer to Figure 3.

4.4 Security Architecture

User account information will be stored in User objects on the server. These will be written to and loaded from a binary file in the server directory.

User passwords will be hashed using the SHA-512 algorithm with a salt length of 24.

When the clients’ user credentials are successfully verified by the server they will gain access to the music player application.

4.5 Communication Architecture

Communication between the client and server will be achieved using sockets.

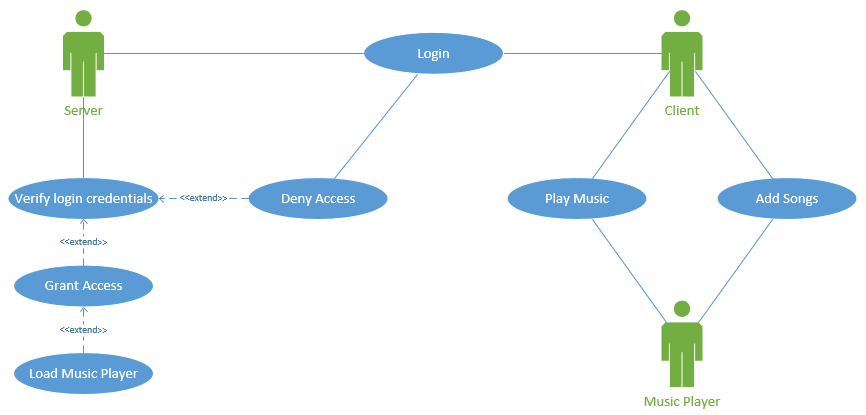
During each login attempt user credential data will be transferred from the client to the server for verification. A response will be returned from the server to the client to notify of successful or unsuccessful login attempt.

4.6 Performance

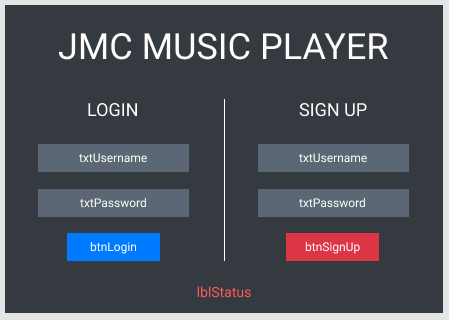
To maximize performance of the application and reduce usage of system resources, all data types must be optimized to store precisely the size of data that each variable is required to hold and all threads must be terminated if no longer required.

# 5 System Design

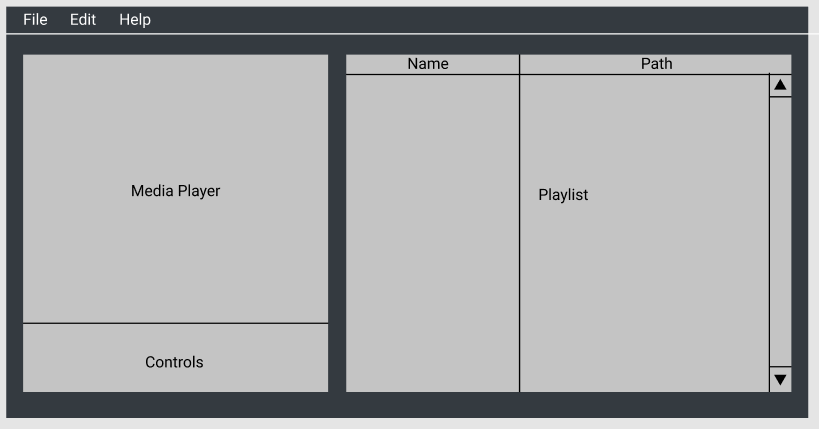
5.1 Use-cases



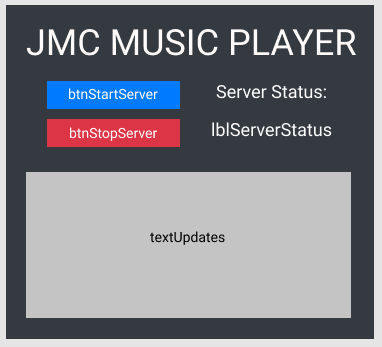
5.2 User Interface Design



Login Form UI Design



Music Player UI Design



Authentication Server UI Design

5.3 System Requirements

Operating System: Windows 10 64-bit

CPU: Intel Core i3

RAM: 4GB

HDD: 256GB

# 6 Implementation Plan

6.1 Goal

The goal of the project is to develop a music player application that meets specific requirements laid out by the client to demonstrate a range of skills and abilities in software development.

6.2 Tasks

1. Complete Pre-Design Specifictation documentation.
2. Complete Product Design Specification documentation.
3. Submit to client for approval.
4. Begin application development.
5. Test all aspects of the application to ensure they meet software requirements.
6. Perform and document unit testing.
7. Produce application help files.
8. Submit completed project to client.

6.3 Resources

Resources required include:

* PC/laptop
* internet connection
* office space

6.4 Responsibilities

Solo development project. All design, documentation, programming and testing by Reece Pieri.

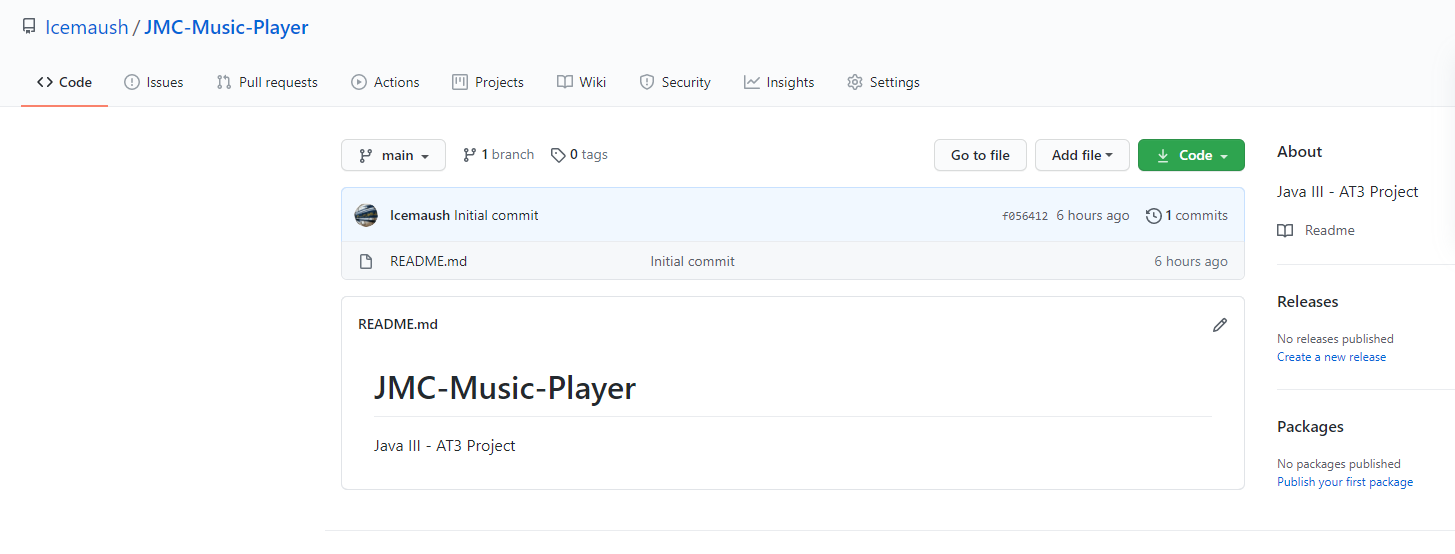
6.5 Success Metrics

Success will be measured against the software requirements to ensure each requirement has been met and the client will be satisfies with the final product.

# 7 Source Control

Source Control will be handled using GitHub. All documentation, source files and testing files will be stored in a GitHub project repository.

GitHub repository: <https://github.com/Icemaush/JMC-Music-Player>



# 8 Test Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **Description** | **Expected Result** | **Actual Result** |
| Case 1 | Start server | Load user information from file and begin listening for incoming connections |  |
| Case 2 | Stop server | Save user information to file and stop listening for incoming connections |  |
| Case 3 | Remove user | Remove a registered user from server |  |
| Case 4 | Login to music player client | Login successful, close login form and open music player client |  |
| Case 5 | Create a new playlist | Successfully create a new playlist |  |
| Case 6 | Add songs to playlist | Add multiple songs to playlist via Open File dialog |  |
| Case 7 | Remove songs from playlist | Song removed from playlist |  |
| Case 8 | Clear playlist | All songs cleared from playlist |  |
| Case 9 | Load playlist | Playlist loaded from CSV file |  |
| Case 10 | Save playlist | Playlist saved to CSV file |  |
| Case 11 | Play music | Song begins playing |  |
| Case 12 | Sort playlist | Merge sort performed on playlist successfully |  |
| Case 13 | Search playlist | Specified song found in playlist |  |
| Case 14 | About | Display About message box |  |

# 9 Product Design Specification Approval

The undersigned acknowledge they have reviewed the **JMC Music Player Product Design Specification** document and agree with the approach it presents. Any changes to this Requirements Definition will be coordinated with and approved by the undersigned or their designated representatives.

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_

Print Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Title: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Role: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_

Print Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Title: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Role: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_