Jyle Darling

M093437

AT3 Product Design Specification

Version 1.0

26/11/2020

# Version History

The development of this application has been control via a GitHub repository, which houses the application code and documentation used. This repository can be found at: <https://github.com/Jely101/Java>

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version #** | **Implemented By** | **Revision Date** | **Approved By** | **Approval Date** | **Reason** |
| 1.0 | Jyle Darling | 26/11/20 | Stewart Godwin | 26/11/20 | Initial Design Definition Draft |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Table of Contents

[Version History i](#_Toc57298158)

[1 Introduction 1](#_Toc57298159)

[1.1 Purpose of the Product Design Specification Document 1](#_Toc57298160)

[2 General Overview and Design Guidelines 1](#_Toc57298161)

[2.1 Standards 1](#_Toc57298162)

[3 Architecture Design 2](#_Toc57298163)

[3.1 Logical View 2](#_Toc57298164)

[3.2 Hardware Architecture 2](#_Toc57298165)

[3.3 Software Architecture 2](#_Toc57298166)

[3.4 Security Architecture 2](#_Toc57298167)

[3.5 Communication Architecture 2](#_Toc57298168)

[3.6 Performance 2](#_Toc57298169)

[4 System Design 3](#_Toc57298170)

[4.1 Use Cases 3](#_Toc57298171)

[4.2 Database Design 3](#_Toc57298172)

[4.3 Data Conversions 4](#_Toc57298173)

[4.4 Application Program Interfaces 4](#_Toc57298174)

[4.5 User Interface Design 4](#_Toc57298175)

[4.6 Performance 5](#_Toc57298176)

[4.7 Section 508 Compliance 5](#_Toc57298177)

[5 Implementation Plan 6](#_Toc57298178)

[5.1 Introduction 6](#_Toc57298179)

[5.2 Specifications and Implementation 6](#_Toc57298180)

[5.3 Code Standards and Testing 7](#_Toc57298181)

[5.4 Summary of Requirements 8](#_Toc57298182)

[6 Test Plan 9](#_Toc57298183)

[7 Product Design Specification Approval 11](#_Toc57298184)

# 1 Introduction

## 1.1 Purpose of the Product Design Specification Document

This Product Design Specification document will document and track 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.

# 2 General Overview and Design Guidelines

This section describes the principles and strategies to be used as guidelines when designing and implementing the system.

## 2.1 Standards

The standards to be used in the design and development of this application are to be in line with those set out by CITE Managed Services.

Naming Conventions

• File Naming and Organization

• Formatting and Indentation

• Comments and Documentation

• Classes, Functions and Interfaces

• Pointer and Reference Usage

• Testing

All systems and projects will be covered by the following ISO Standard(s)

ISO/IEC/IEEE 12207:2017 Systems and software engineering — Software life cycle processes

# 3 Architecture Design

This project as a whole is made up two separate application designed to interact with each other as well as the user’s computer. The client side and server side applications interact with each other through a socket connect, when verifying login credentials. The client side application interacts with the user’s computer through the loading and saving of CSV files.

## 3.1 Logical View

Information regarding the Logical View of the Project can be found in the publically accessible GitHub repository: <https://github.com/Jely101/C-project>

## 3.2 Hardware Architecture

Information regarding the Hardware Architecture of the Project can be found in the publically accessible GitHub repository: <https://github.com/Jely101/C-project>

## 3.3 Software Architecture

Information regarding the Software Architecture of the Project can be found in the publically accessible GitHub repository: <https://github.com/Jely101/C-project>

## 3.4 Security Architecture

Information regarding the Security Architecture of the Project can be found in the publically accessible GitHub repository: <https://github.com/Jely101/C-project>

## 3.5 Communication Architecture

Information regarding the Communication Architecture of the Project can be found in the publically accessible GitHub repository: <https://github.com/Jely101/C-project>

## 3.6 Performance

Information regarding the Performance of the Project can be found in the publically accessible GitHub repository: <https://github.com/Jely101/C-project>

# 4 System Design

## 4.1 Use Cases

|  |
| --- |
| Name: Open and load music player  Description: User to login in and open music player to load a music playlist.  Preconditions: User’s data is already entered into the system. Playlist file previously saved on users computer.  Post conditions: music player loaded and music playing.  Basic Course of Action:   1. User opens media player application. 2. User logs in to server with validated credentials. 3. Server application verifies login credentials and returns access granted signal 4. Media player application opens. 5. User opens a playlist file stored on their computer via open file dialog contained in menu bar. 6. Playlist file opens and populates data into the application and media player playlist. 7. User presses play button. 8. Music plays. |

|  |
| --- |
| Name: Add a new user.  Description Server application to add new user.  Preconditions: Server application is running.  Post conditions: New user created.  Basic Course of Action:   1. User enters new unique username. 2. User enters desired password. 3. User selected Add new user button. 4. Server application verifies entered information is eligible. 5. Server application hashes password and adds information to a class object. 6. Server application adds class object to a list 7. List objects sorted by username. 8. Message box displayed showing new user successfully added. |

## 4.2 Database Design

No databases were used in the creation of this application, though the storing of Username data could be substituted for a database table.

## 4.3 Data Conversions

No data conversions are used within this application

## 4.4 Application Program Interfaces

Information regarding the Application Program Interfaces of the Project can be found in the publically accessible GitHub repository: <https://github.com/Jely101/C-project>

## 4.5 User Interface Design

The following mock up designs of the UI are as follows:

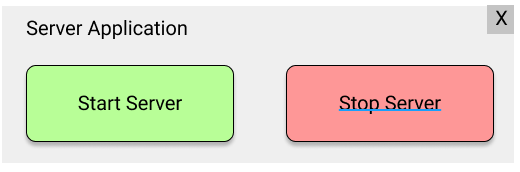


Figure 1 Mock-up of the server application GUI

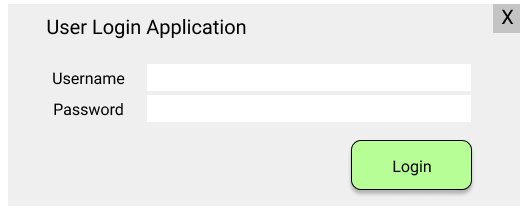


Figure 2 Mock-up of the user Login window

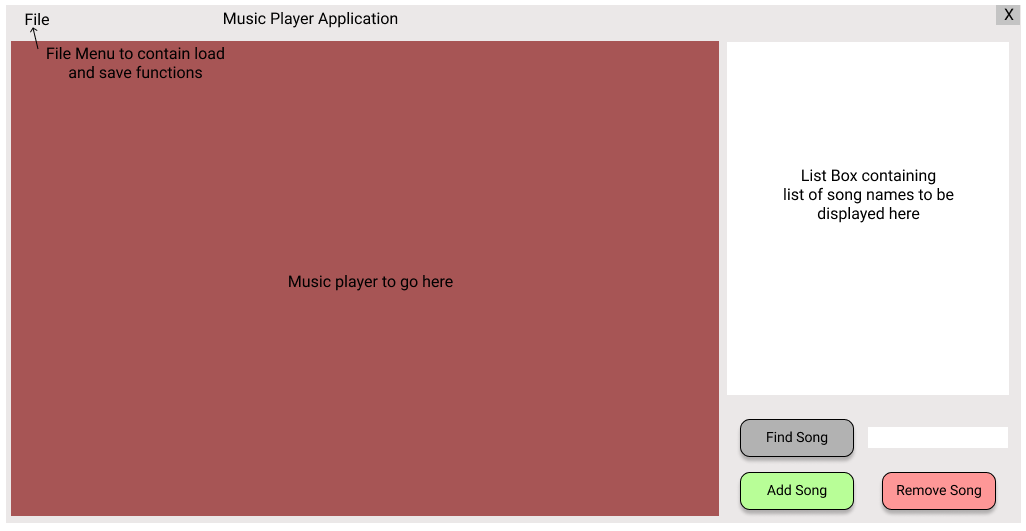


Figure 3Mock-up of the user media player application.

## 4.6 Performance

Information regarding the Performance of the Project can be found in the publically accessible GitHub repository: <https://github.com/Jely101/C-project>

## 4.7 Section 508 Compliance

Information regarding the Compliance of the Project can be found in the publically accessible GitHub repository: <https://github.com/Jely101/C-project>

# 5 Implementation Plan

## 5.1 Introduction

This document will outline the basic requirements of the proposed application and how these requirements will be implemented or achieved within the application.

The application itself is an advanced music player that provides the ability to sort and search the songs SortedSet, the GUI should display the sorted track list and highlight and play the searched track, it should save the track list to a csv using a 3rd party library. The music player must also load and play files.

The above requirements have been used as a foundation for a number of other requirements that have been specified for this application, and a more descriptive plan for the application will be listed below.

## 5.2 Specifications and Implementation

Although this document will focus mainly on a client application, a secondary application will also be developed in conjunction to help facilitate some of the additional requirements. This will be a server client relationship between these two applications.

The client application will be comprised of a basic login feature, a media player, a way to sort, search, add and remove songs to/from a list of song names stored, and a method to load and save the song list.

The server application will facilitate the login and authentication of the login credentials, which will unlock the client’s application to give access to all the features.

The client application will begin with a login function. The login in details will be sent to the server application, where the password will be hashed and verified against a list of stored objects that contain a username, hashed password and salt associated with the hashed password. Once the server verifies the login in credentials, a signal will be sent back to the client application which will open a new windows form, that contains the music player connected functions.

The music player application will contain a menu bar that allows for a song list to be saved and loaded.

* When a song list is saved, it will be stored as a CSV File containing the song names and file path locations. This CSV document will be stored on the user’s computer.
* When a song list is loaded, the data will be retrieved via a third party library. Once the CSV data has been retrieved the items will be added to a sorted set.

Once a music list has been loaded, the data contained in the sorted set will be displayed as a list view within the application. Songs double clicked in the list view will be played by the music player. Below the list view box there will be a text box and button to facilitate the searching of the sorted set, a button to add a new song, and a button to remove a song from the list.

* Song names will be searched in the sorted set using a linear search method, if a song is found in the set, it will be highlighted in the list view, otherwise a message box will display if the item is not found.
* Adding a song to the list will first open an open file dialogue, where the user can select a single or multiple files to be added to the list. Once added, the songs will be added to the sorted set, which will populate the list view.
* Removing an item from the song list will remove it from the sorted set, and repopulate the list view.

The Server application will be quite simple, the GUI will contain two buttons, one to start the server and another to stop it. The server application will contain a list that contains class objects that save user information, such as username, hashed password and salt used with the hashing process. Class objects that are added to this list will be sorted using a merge sort based on the class object property ‘username’.

## 5.3 Code Standards and Testing

All documentation and code files will be uploaded to GitHub throughout the development of this application. The code repository for this application can be found at <https://github.com/Jely101/C-project>.

A list of the code standard being enforced are:

* Naming Conventions
* File Naming and Organization
* Formatting and Indentation
* Comments and Documentation
* Classes, Functions and Interfaces
* Pointer and Reference Usage
* Testing

These standards how been outlined in:

[ISO/IEC/IEEE 12207:2017 Systems and software engineering — Software life cycle processes](https://www.iso.org/standard/63712.html)

## 5.4 Summary of Requirements

This is a quick summary of the application requirements and their location within the application.

**Dynamic Data Structures:**

* Server application contains a List of Class objects storing user data for login authentication.
* Client application contains a SortedSet which contains the song file path strings.

**Hashing Technique:**

* Server application contains functions which hashes created username passwords with salt and stores them in class objects alongside username’s and associated salt.

**Sorting Algorithm:**

* Server application contains a Merge Sort to sort the list of user data object by the username property.

**Searching Algorithm:**

* Client application contains a linear search that iterates through the SortedSet to try and find the specified song name.

**Third Party Library:**

* Client application used a third party library to facilitate the reading of CSV files and populating the file data into the SortedSet.

**Application GUI:**

* Both client and server applications are WinForm applications providing a GUI for users.

**Code Standards:**

* Both applications adhere to code standard set out in this document.

# 6 Test Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case** | **Description** | **Expected Result** | **Actual Result** | **Comments** |
| 01. | Server - Creating a new user in server application | User details added to list via a class object, message box to display showing user was added successfully. |  |  |
| 02. | Server - Starting server socket connection | Pressing the server on starts the server connection, button turns green. |  |  |
| 03. | Server - Stopping the server connection | Pressing the server off button stops the server connection and button turns red |  |  |
| 04. | Client – Login attempt when server not connected. | Message to show connection issue. |  |  |
| 05. | Client – Login attempt invalid credentials while server is connected. | Message to show login failed due to invalid credentials. |  |  |
| 06. | Client – Login attempt valid credentials server connected. | Media Player application to load |  |  |
| 07. | Client – Opening CSV file containing song playlist via menu bar item. | Menu bar item to open an open file dialog, selecting the CSV file will populate the List box with file contents. Songs will play if selected. |  |  |
| 08. | Client – Adding new songs to the playlist | Add new song button will open an open file dialog allowing one or more files to be added to the playlist. |  |  |
| 09. | Client – playing a song by double clicking the item in the list box | Double clicking an item in the list box will cause it to be played in the media player |  |  |
| 10. | Client – Removing a song from the list box | Selecting a song in the list and pressing remove selected song button will remove the song from the list. |  |  |
| 11. | Client – Removing a song from the list box with nothing selected. | Pressing the remove selected song button with no list item selected will not do anything |  |  |
| 12. | Client – Searching for a song in the list | Searching for a song in the list will highlight the song and cause it to start playing |  |  |
| 13. | Client – Searching for a song not in the list | Nothing will be highlight or played by the media player |  |  |
| **Test Case** | **Description** | **Expected Result** | **Actual Result** | **Comments** |
| 14. | Saving a playlist to a csv file | Menu bar item will open a save file dialog which allows the contents of the list box to be saved to a CSV file |  |  |
| 15 | Loading recently saved file to ensure data was saved correctly | Menu bar item to open an open file dialog, selecting the CSV file will populate the List box with file contents. Songs will play if selected. |  |  |
|  |  |  |  |  |

# 7 Product Design Specification Approval

The undersigned acknowledge they have reviewed the *AT3* **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: | J. Darling | Date: | 26/11/20 |
| Print Name: | Jyle Darling |  |  |
| Title: | Project Manager |  |  |
| Role: | Lead Project Manager |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Signature: |  | Date: | 26/11/20 |
| Print Name: | Ken Beck |  |  |
| Title: | Client / Business Representative |  |  |
| Role: |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Signature: |  | Date: |  |
| Print Name: |  |  |  |
| Title: |  |  |  |
| Role: |  |  |  |