# Table of Content

Contents

[Table of Content 1](#_Toc86049418)

[1. Introduction 2](#_Toc86049419)

[1.1 Purpose 2](#_Toc86049420)

[1.2 Intended audience and reading suggestion 2](#_Toc86049421)

[1.3Project scope 2](#_Toc86049422)

[1.4 references 2](#_Toc86049423)

[2. Overall Description 2](#_Toc86049424)

[2.1 Product Perspective 2](#_Toc86049425)

[2.2 Product Features 2](#_Toc86049426)

[2.3 User Class and Characteristics 2](#_Toc86049427)

[2.4 Operating Environment 2](#_Toc86049428)

[2.5 Design and Implementation Constraints 2](#_Toc86049429)

[3. System Features 2](#_Toc86049430)

[3.1 Description and Priority 2](#_Toc86049431)

[3.2 Functional Requirements 3](#_Toc86049432)

[4. External Interface Requirements 3](#_Toc86049433)

[4.1 User Interfaces 3](#_Toc86049434)

[4.2 Hardware Interfaces 3](#_Toc86049435)

[4.3 Software interfaces 3](#_Toc86049436)

[5. Nonfunctional Requirements 3](#_Toc86049437)

[5.1 Performance requirements 3](#_Toc86049438)

[5.2 Safety Requirements 4](#_Toc86049439)

[5.3 Security Requirements 4](#_Toc86049440)

# Introduction

## Purpose

The purpose of this project is to build an advance music player that allows the ability to sort and search the songs stored in a singly linked list but before accessing the music player JMC employee need to log in using the provided username and password by JMC. It will also write out the song play list once the application closes using a third party CSV writer/reader.

## 1.2 Intended audience and reading suggestion

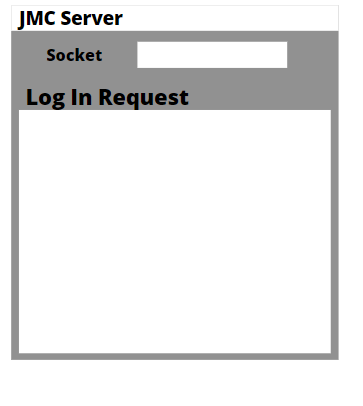
This project is a test program for Jupiter Mining Corporation and is design to be an advance music player for their employee. I will be using a client/server program to allow JMC employee to log in to access the music player program.

## 1.3Project scope

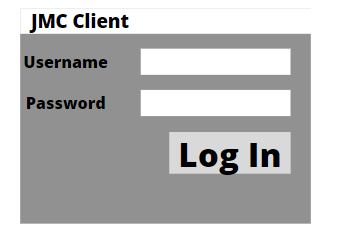
The purpose of the advance music player is to demonstrate my range of skills and abilities to JMC, the advance music player will allow user to have the ability to sort and search songs stored in a doubly linked list and there will be a verification process eg a log in system that uses client/server program to communicate.

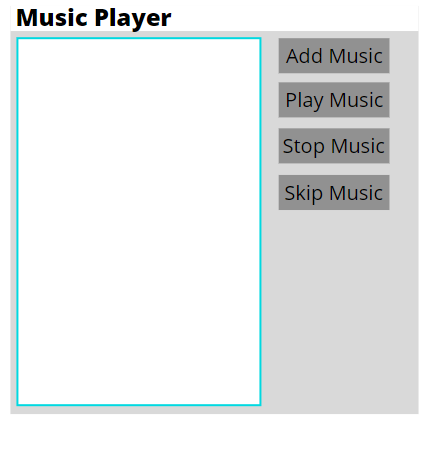
## 1.4 GUI Prototype

Server Side



JMC Client log in





# Overall Description

## 2.1 Product Perspective

An advance music player that stores the songs in a dynamic data structures which utilize a merge sort to organize the songs and have a built in binary search. A verification process through log in system using client/server program to communicate.

## 2.2 Product Features

The product features are the ability to utilize client/server program to stored songs in a dynamic data structure, merge sort, binary search, stop songs, play songs and to print a songs list.

## 2.3 User Class and Characteristics

The users of the system should be able to add songs, search songs, play and stop the songs after logging in using the provided username and password from JMC

## 2.4 Operating Environment

Operating environment for the advance music player is as listed below

Operating system: Windows/Linux/MacOS  
 Platform: Java

## 2.5 Design and Implementation Constraints

Required operating system windows in order to run

# System Features

## 3.1 Description and Priority

An advance music player that allows the users to sort and search the songs stored in a binary tree, the GUI should display the sorted track list and highlight and play the searched track, it should also save the track list to a csv using a 3rd party library.

## 3.2 Functional Requirements

1. Add Songs

2. Store in a binary tree

3. Sort Songs

4. Search Songs

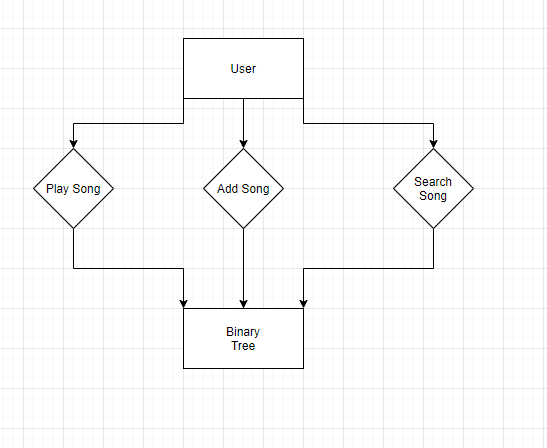
5. Have a GUI

6. Save track list to a CSV

# Nonfunctional Requirements

## Performance requirements

This program will allow the user to add, search and play songs. Also it will have a basic objective of normalization in order to reduce redundancy which means that information is to be stored only once, storing information several times leads to wastage of storage space and increase in the total size of the data stored.



## Safety Requirements

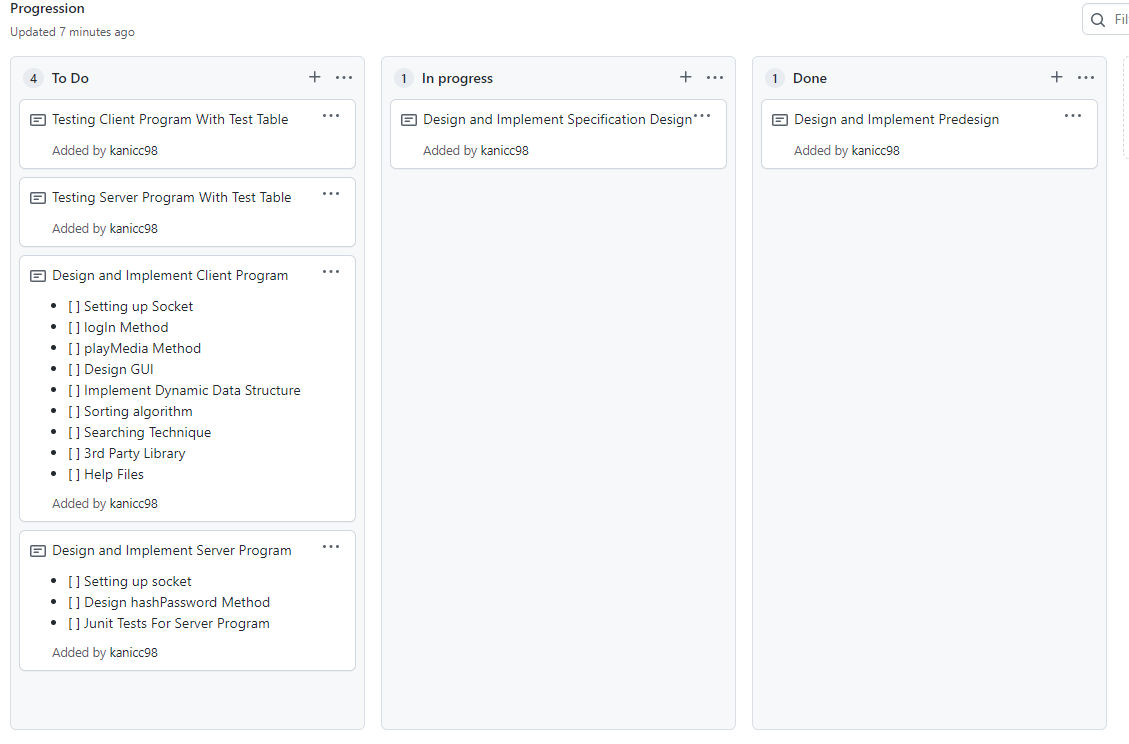
If there is crash in run time the csv will write out all the list of songs that have previously been added and once the user load the program again it will automatically add the songs.

## Security Requirements

The locations of the songs will be hash once the user add the songs into the list.

# 5. Implementation plan

For implementation plan I choose to use GitHub built in projects Kanban board as it act like a collaboration software and an implementation plan at the same time.



## 5.1 Version Control

Version control also known as source control is the practice of tracking and managing changes to software code which GitHub is able to fulfill all the requirements for a version control software therefore I will be using GitHub as my version control software

https://github.com/kanicc98/Java\_MusicPlayer

# Test Plan

## Test Table

For this test table it is important that we test all the functional requirements that will be implemented

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case | Input | Expected Outcome | Actual Outcome | Comment |
| hashPassword |  |  |  |  |
| logIn |  |  |  |  |
| playMedia |  |  |  |  |
| stopMedia |  |  |  |  |
| skipMedia |  |  |  |  |
| 3rd party csv writer |  |  |  |  |

## Unit Testing

For Unit testing it is important to test all the functions with the server/client programs this include hash password, socket connections, log In , play media, stop media, skip media, add media, 3rd party csv reader/writer, sort function and finally search function.