SOFTWARE REQUIREMENTS SPECIFICATION

**For**

**Music Library Management**

**Prepared by:-**

*TEAM 12*

*Shrivarshan C*

*Waseef Tauqueer A*

*Sudharsan k k*

# Introduction

## Purpose

## The primary goal of this document is to outline the requirements for the Music Library Management System project. It provides a comprehensive description of both functional and non-functional requirements as proposed by the client. The project aims to create a user-friendly environment for accessing and streaming music. This project describes the software interface necessities employing ER diagrams and UML diagrams.

## Document Conventions

* + - Entire document should be justified.
    - Convention for Main title

Font face: Times New Roman Font style: Bold

Font Size: 14

* + - Convention for Sub title

Font face: Times New Roman Font style: Bold

Font Size: 12

* + - Convention for body

Font face: Times New Roman Font Size: 12

## Scope of Development Project

## The Music Library Management System functions as a comprehensive music streaming platform, facilitating user interaction through features such as playlist management and favoriting specific tracks. It serves as an intuitive user interface, enabling seamless modification of playlists, albums, and preferred songs tailored to individual user preferences.

## This adaptable system has versatility across a range of scenarios, accommodating the incorporation of new functionalities as needed. Its modular design encourages reusability and scalability, ensuring flexibility across all modules. Java was selected as the development language due to its exceptional performance, extensive toolset, cross-platform compatibility, rich libraries, cost-effectiveness (being freely available), and streamlined development process

## Definitions, Acronyms and Abbreviations

JAVA -> platform independence SQL-> Structured query Language ER-> Entity Relationship

UML -> Unified Modeling Language

IDE-> Integrated Development Environment SRS-> Software Requirement Specification

## References

* + - Books

 Software Requirements and Specifications: A Lexicon of Practice, Principles and Prejudices (ACM Press) by Michael Jackson

Software Requirements (Microsoft) Second EditionBy Karl E. Wiegers

Software Engineering: A Practitioner’s Approach Fifth Edition By Roger S. Pressman

* + - Websites

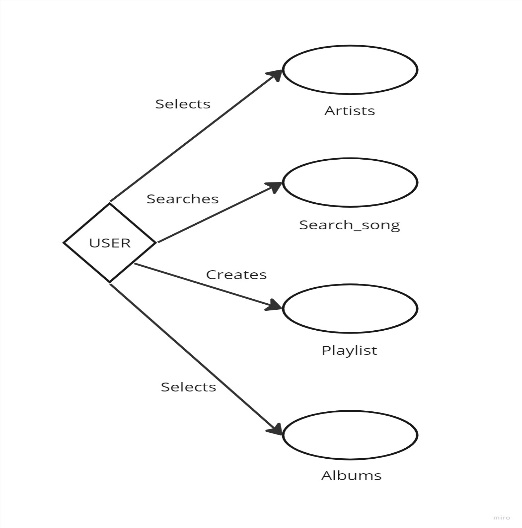
[**http://www.slideshare.net/**](http://www.slideshare.net/)

[**http://ebookily.net/doc/srs-library-management-system**](http://ebookily.net/doc/srs-library-management-system)

# Overall Descriptions

## Product Perspective

Use Case Diagram of Music Library Management System

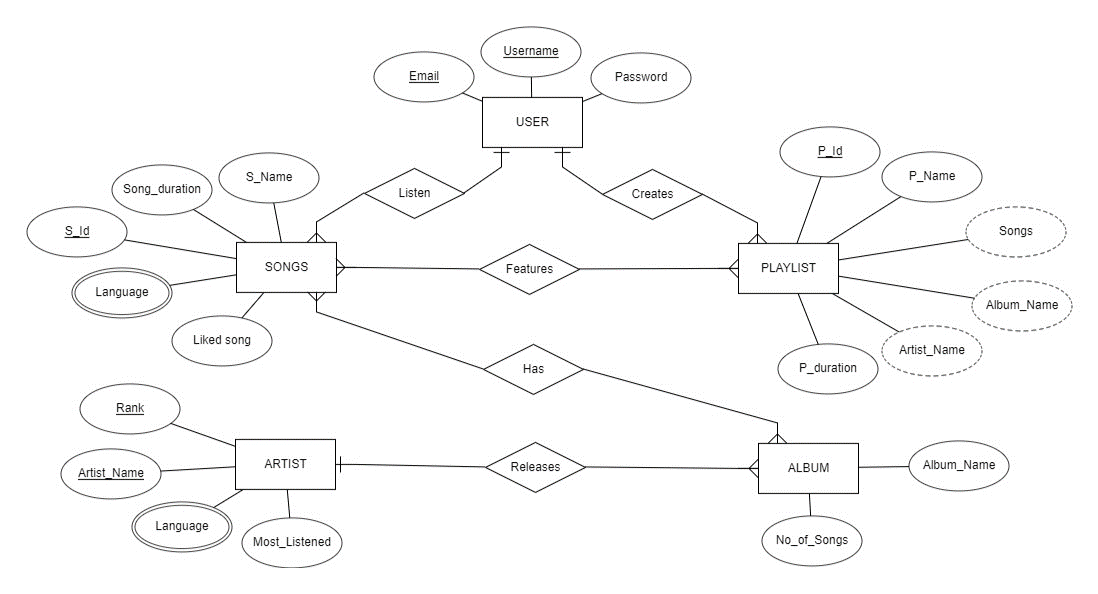


This diagram provides an overview of the Music Library Management System tailored for users, offering functionalities catering to music enthusiasts. Users, categorized as listeners, can utilize the system's search feature to explore music resources based on song titles, artists, albums, or genres.

Within the system, listeners can personalize their music experience by creating playlists, marking favorite tracks, and downloading music based on their preferences. The system allows users to manage the player, playlists, organize favorite songs, search and access download options following specific guidelines or criteria established within the system

## Product Function

Entity Relationship Diagram of Music Library Management System.



The Music Library System is a versatile platform designed to cater to diverse musical tastes and requirements. Through its focus on playlists tailored to individual moods or genres, users have the freedom to curate and modify tracks, share their compilations, and easily access personalized collections. The 'Liked Songs' feature provides a quick repository for favorite tracks, streamlining accessibility. By organizing music according to artists, users can explore comprehensive discographies, follow preferred musicians, and stay updated on releases or concerts. Additionally, the system facilitates browsing music by albums, empowering users to immerse themselves in their favorite artists' complete bodies of work or discover new genres and albums of interest. This multifaceted system offers a rich, personalized, and engaging musical experience.

## User Classes and Characteristics

**Administrator:**

1. Issue Music Albums:

- Ability to issue music albums to members.

2. View Categories:

- Access to view different music genres or categories available in the system.

3. View Album List:

- Ability to view the list of music albums available in each category.

4. Return Music Albums:

- Capability to receive returned music albums from members.

5. Add Albums to Database:

- Authority to add new music albums to the database, including information like artist, title, genre, release date, etc.

6. Edit Album Information:

- Capability to edit the information of existing music albums in the database.

7. Access Member Accounts:

- Access to all member accounts for management purposes.

**Member (User):**

1. View Categories:

- Access to view different music genres or categories available in the system.

2. View Album List:

- Ability to view the list of music albums available in each category.

3. Own Account:

- Capability to create and own an account in the system.

4. View Issued Albums:

- Ability to view the list of music albums issued to the member.

5. Request New Albums:

- Capability to put in a request for a new music album to be added to the system.

6. View Issued History:

- Access to view the history of music albums issued to the member previously.

7. Search for Albums:

- Capability to search for a particular music album in the collection.

These features provide a structured framework for the Music Management System, ensuring that both administrators and members have the necessary tools to manage and access the music resources effectively.

## Operating Environment

The product will be operating in windows environment. The Music Library Management System is a desktop application and shall operate in desktops and all famous browsers. The only requirement to use this online product would be the internet connection.

The hardware configuration includes Hard Disk: 40 GB, Monitor: 15” Color monitor, Keyboard: 122 keys. The basic input devices required are keyboard, mouse and output devices are monitor, etc.

## Assumptions and Dependencies

**Music Library Management System Assumptions:**

* + - Flawless code: No bugs or crashes to disrupt your music flow.
    - Easy navigation: Like a breeze through your favorite tunes.
    - Seamless storage: Songs always there, ready to play, from anywhere.
    - Speedy search: Find that perfect track in a flash.
    - 24/7 access: Music awaits, whenever you need a fix.

The dependencies are:-

* + - JavaFX
    - Servers, JavaFX, music, cloud - the tech orchestra.
    - Roles, features, interface, security - the sheet music for how it works.
    - Admin training, reports, backups - keeping the JavaFX stage organized and data understood.
    - User info, music details, updates - all stored accurately and securely, backstage.
    - Scalability, performance, mobile access - the JavaFX app adapts to your growing audience.
    - Backup plan - because even music needs an understudy.

## Requirement

Software Configuration: -

This software package is developed using javafx as front end which is supported with SceneBuilder. Microsoft SQL Server as the back end to store the database.

Operating System: Windows 11

Language: Java, Java fxml

Database: MS SQL Server (back end)

Hardware Configuration: -

Processor: Windows 7 or more

Hard Disk: 100GB

RAM: 4gb or more

## Data Requirement

The inputs consist of the query to the database and the output consists of the solutions for the query. The output also includes the user receiving the details of their accounts. In this project the inputs will be the queries as fired by the users like create an account, selecting songs, selecting albums, selecting artists and creating playlist. Now the output like playlist, favorite songs and albums will be visible in the viewer side once each operation is completed.

# External Interface Requirement

## GUI

The software provides good graphical interface for the user to operate on the music system, performing tasks such as listening to music, creating playlist, selecting favorites, viewing artists and albums.

* + - It allows user to view the songs and play what they wish to.
    - The users can play the song by finding them through search option.
    - It includes playlist facility where the user can create their own playlists.
    - All the features like viewing albums, songs according to artists and favorites are included.
    - The design should be simple and all the different interfaces should follow a standard

template.

Login Interface: -

In case the user is not yet registered, they can enter the details and register to create their account. Once the account is created, they can ‘Login’ which asks the user to type his username and password. If the user entered either his username or password incorrectly then an error message appears.

Search: -

The user can search for songs according to their interests through the search feature.

Playlist: -

Users can create playlists including all the songs they like.

Listen: -

The users can listen to the song from playlist, from selected artists, albums and favorites.

# System Features

# Secure Music Library with JavaFX:

# Login: Username/password + MFA for extra protection.

# Admin monitoring: Track activity, set limits, flag suspicious behavior.

# User isolation: Each account is a private music island.

# Admin control: Full access for managing accounts and activity.

# JavaFX bonus: Visual indicators, popups, and admin dashboards.

# Fines (optional): Design a JavaFX interface for fine management.

# Reports and analytics: Insights into user behavior and music trends.

# Security focus: Regular updates, user education, and data privacy compliance

# Other Non-functional Requirements

## Performance Requirement

The performance of the offline music library system hinges on multiple facets. It necessitates lightning-fast responses from its database, ensuring seamless navigation and operations even with vast music collections. Swift offline access and judicious storage usage are imperative for a smooth user experience, allowing efficient handling of metadata, playlists, and searches without compromising speed. Compatibility across various devices and platforms is pivotal, ensuring a consistent, quick, and hassle-free experience. Robust backup, synchronization capabilities, and a highly responsive user interface are pivotal components, guaranteeing swift operations devoid of any discernible delays or lags, thereby elevating the user's interaction with the system.

## Safety Requirement

The database may get crashed at any certain time due to virus or operating system failure. Therefore, it is required to take the database backup so that the database is not lost. Proper UPS/inverter facility should be there in case of power supply failure.

## Security Requirement

* The Music Library Management System prioritizes robust security measures to protect sensitive information and ensure the system's integrity.
* Utilizing encryption, the database securely stores music-related data.
* User access is carefully controlled, permitting normal users to read information with restrictions on editing, except for personal details.
* A role-based access system categorizes users into admin and regular roles, each with specific access levels.
* Robust authentication, including multi-factor authentication, safeguards user identities, while stringent password policies and secure storage methods prevent unauthorized access.
* Protective measures against hacking, such as account lockouts, are in place. Admin controls are segregated, granting exclusive database update rights to administrators.
* An audit trail logs user activities, facilitating regular monitoring for potential security issues. Regular data backups, stored securely with a recovery plan, mitigate the risk of data loss.
* Additionally, user security training promotes awareness and adherence to best practices, collectively ensuring a secure and resilient Music Library Management System.

## Requirement attributes

* + - There may be multiple admins creating the project, all of them will have the right to create changes to the system. But the members or other users cannot do changes
    - The project should be open source
    - The Quality of the database is maintained in such a way so that it can be very user friendly to all the users of the database
    - The user be able to easily download and install the system

## Business Rules

In the context of the Music Management System, a business rule encompasses guidelines and practices that capture and implement policies specific to the music industry. These rules serve to enforce business policies, make decisions, or derive new data from existing information. They extend to regulating aspects such as project costs and discount offers within the system. Users are expected to adhere to these rules and regulations, avoiding any engagement with illegal practices or protocols. Both administrators and members are obliged to stay within the defined boundaries of the system's rules and regulations, ensuring compliance and integrity.

## User Requirement

The Music Management System caters to both music lovers and tech-savvy admins. Users with basic computer skills can effortlessly navigate the system's user-friendly interface, while comprehensive manuals and online resources offer deeper support. Admins, like librarians for the musical realm, ensure data security through backups, disaster recovery, and clever data replication strategies. They keep the system humming smoothly with regular maintenance, updates, and a keen eye on file organization. This synergy between user convenience and administrative expertise creates a harmonious music haven, where melodies and data exist in perfect equilibrium.

# Other Requirements

## Data and Category Requirement

In the Music Library Management System, users are classified into different categories. Access rights are determined by the user category. Administrators have the privilege to modify, delete, and append data, while all other users are restricted to retrieving information from the database.

Similarly, the music library encompasses various categories of music. Each category's relevant data should be appropriately displayed, and the categories along with their associated data must be coded in a specific format. This ensures that users can easily access and interact with the music library based on their roles and preferences.

## Appendix

A: Admin, Abbreviation, Acronym, Assumptions; B: Books, Business rules; C: Class, Client, Conventions; D: Data requirement, Dependencies; G: GUI; K: Key; L: Library, Librarian; M: Member; N: Non-functional Requirement; O: Operating environment; P: Performance, Perspective, Purpose; R: Requirement, Requirement attributes; S: Safety, Scope, Security, System features; U: User, User class and characteristics, User requirement;

## Glossary

The following are the list of conventions and acronyms used in this document and the project as well:

* + - Administrator: A login id representing a user with user administration privileges to the software
    - User: A general login id assigned to most users
    - Client: Intended users for the software
    - SQL: Structured Query Language; used to retrieve information from a database
    - SQL Server: A server used to store data in an organized format
    - Layer: Represents a section of the project
    - User Interface Layer: The section of the assignment referring to what the user interacts with directly
    - Application Logic Layer: The section of the assignment referring to the Web Server. This is where all computations are completed
    - Data Storage Layer: The section of the assignment referring to where all data is recorded
    - Use Case: A broad level diagram of the project showing a basic overview
    - Class diagram: It is a type of static structure diagram that describes the structure of a system by showing the system’s cases, their attributes, and the relationships between the classes
    - Interface: Something used to communicate across different mediums
    - Unique Key: Used to differentiate entries in a database

## Class Diagram

A class is an abstract, user-defined description of a type of data. It identifies the attributes of the data and the operations that can be performed on instances (i.e. objects) of the data. A class of data has a name, a set of attributes that describes its characteristics, and a set of operations that can be performed on the objects of that class. The classes’ structure and their relationships to each other frozen in time represent the static model. In this project there are certain main classes

which are related to other classes required for their working. There are different kinds of relationships between the classes as shown in the diagram like normal association, aggregation, and generalization. The relationships are depicted using a role name and multiplicities. Here ‘User’, ‘Songs’, ‘Playlist’, ‘Artist’ and ‘Album’ are the most important classes which are related to other classes.

