SOFTWARE REQUIREMENTS SPECIFICATION

**For**

**Music Library Management System**

**Prepared by:-**

*Joshika R*

*Kaviya R*

*Monica M*

# 1.Introduction

## Purpose

The main objective of this document is to illustrate the requirements of the project Music Library Management system. The document gives the detailed description of the both functional and non-functional requirements proposed by the client. The purpose of this project is to provide a friendly environment to maintain their song playlist. The main purpose of this document is to outline the requirements for a Music Library Management System. This project is to categorize and catalog every single piece of music info and reviews, rate and review albums, online and offline music service, listing record collections, music downloading and sharing etc. This project describes the hardware and software interface requirements using 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

A Music Library Management System basically allows users to organize, categorize,

and manage their music collection in a user-based application that allows users to know more

about their accounts, playlists, and top artists.

This project is specifically designed to organize music according to the user's way of

thinking. This product serves as a complete user interface for the music library management process

and music classification. Music Library Management Systems can be used through existing interfaces

to create, insert, and monitor playlists of songs. It's especially convenient for users to be able to listen

to their favourite songs no matter where they are in the world.

The project can be easily implemented in a variety of situations. New features can be added

as needed, and all modules are flexible and reusable. The language used to develop the project is Java.

Java has significant advantages over other languages ​​in terms of performance, available tools,

cross- platform compatibility, libraries, cost (it's freely available), and development process.

## Definitions, Acronyms and Abbreviations

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

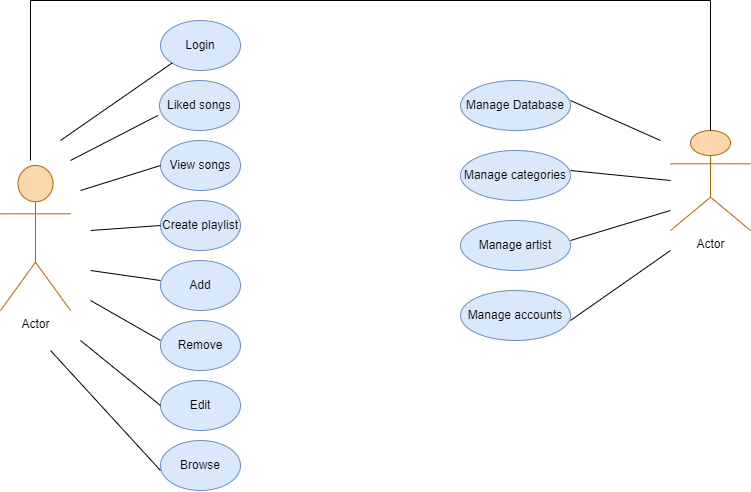
UML -> Unified Modeling Language

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

# 2.Overall Descriptions

## 2.1Product Perspective

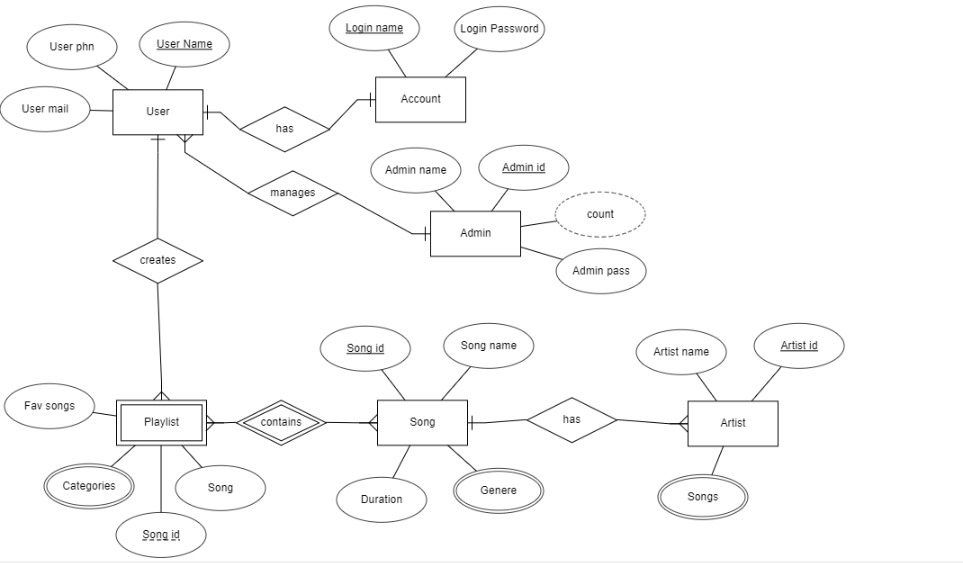
Use Case Diagram of Music Library Management System



This is a broad level diagram of the project showing a basic overview. This System will provide a search functionality to facilitate the search of songs. This search will be based on various categories viz artist name or keyword. Further the each individual personnel can add/update the songs users from the system. The users of the system can request view/download/like the music for which they would have to follow certain criteria.

**2.2 Product Function**

Entity Relationship Diagram of Music Library Management



A Music Library Management System is a software application designed to organize, catalog,

and manage a collection of music files or records. The primary goal of such a system is to provide

users with efficient ways to browse, search, and enjoy their music collection. The system maintains

a comprehensive catalog of music entities, including artists, albums, and songs. Users can search

for specific artists, albums, or songs based on various criteria. By selecting songs, users can create

their own playlist of their favorite songs. Recommendations based on listening history and user

preferences might be provided by the system. Users can narrow down their searches by release

date, genre, or other criteria by using the filter options**.**

## 2.3 User Classes and Characteristic:

The system offers various podcasts and music genres according to the user's kind.

In addition to overseeing the playlists, the administrator will have all administrative rights.

The features that are available to the Users are:

* + - Anywhere in the world, a user can listen to podcasts or songs in many languages.
    - Can create different playlist of songs according to user’s wish
    - Can view the list of liked songs in my library.
    - Possibility to make custom playlists with songs based on user preferences.
    - In case they are unsure of the precise music, they can look it up using keywords.
    - Can view the listening history in settings
    - Can download songs for listening offline
    - Can link with your friend’s account and listen the songs together.
    - Multiple users can listen to the same songs simultaneously and enjoy it

The features that are available to the Agents are:

* + - Can upload the different categories of music or podcasts
    - Can view the List of songs where more people are interested to hear
    - Can own an account in this app or software.
    - Can view the songs and lyrics uploaded by agents
    - Can analyze the performance of songs released

# 3 External Interface Requirement

## 3.1 GUI:

## The software offers a user-friendly graphical interface that allows both administrators and users to efficiently operate the system and accomplish necessary tasks like creating and updating playlists and viewing their details.

* + - The application enables users to listen to songs in various languages worldwide.
    - It provides a song list based on the artists whose songs are most played, and its search function is user-driven.
    - The administrator needs to be able to customize the user interface.
    - Every software module must be compatible with this graphical user interface and adhere to the established standards.
    - All the interfaces should adhere to a standard and the design should be kept simple model
    - A portion of the user interface needs to be devoted to the login/logout module, and it should be able to communicate with the user management module.

Login Interface:

If the user hasn't registered yet, he/she can do so by entering his information and starting the registration process. After creating an account, he/she can select "Login," which prompts the user to enter his/her password and username. An error message appears if the user entered their password or username incorrectly.

Search:

The user can specify the kind of music they want to listen to as well as the artist they like, and then

they can search for the specific song by typing in the name of the song.

Categories View:

The categories view allows the user to add, edit, and remove songs from the playlist in addition

to displaying the available song categories.

User Control Panel:

The user can add, edit, or remove playlists and add/remove songs using this control panel.

# 4.System Features

# For a music library management system to effectively arrange, classify, and oversee a sizable music library, it must have a number of features. The following are some crucial components of a music library management system:

* Authenticate users to the system to guarantee safe entry.
* Create an intuitive user interface for quick navigation and effective music file management.
* Add views specifically for albums and artists to give a comprehensive overview of the music collection.
* Permit users to customize playlists and manage them as they see fit

.

# 5. Other Non-functional Requirements

## 5.1Data and Category Requirement

Managing data and classifying it well in a music library management system is essential to

giving users a smooth and well-organized experience. Keep a variety of audio file formats

on hand, including FLAC, WAV, and MP3. Provide tools for importing audio files from

various sources. Construct and oversee an extensive collection of musical genres. Permit

users to tag specific tracks with genres. Permit users to classify or folder-organize their

playlists encourage the development of playlists with themes or varying moods.

## 6.4 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.

## 