**SOFTWARE**

**REQUIREMENTS SPECIFICATION**

**For**

# Music Library Management System

**Prepared by:-**

**Swetha.S**

**ShanmugaPriya.M**

**Gnanariddhika.R**

## 1. Introduction

### 1.1 Purpose

The primary goal of this document is to outline the project requirements for the Music Library Management System. The document provides a comprehensive overview of requirements as specified by the client. The objective of this project is to create a user-friendly platform for organizing music and managing library users. The Music Library Management System aims to provide an accessible and efficient platform for managing music collections and user profiles within a library setting. The key focus is on establishing a seamless system for music circulation through computerized means and generating various reports. The document further elucidates the software interface requirements through the utilization of ER diagrams and UML diagrams.

**1.2 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

### 1.3 Scope of Development Project

The Music Library Management System is a cutting-edge initiative, transforming the traditional music library into a dynamic web-based application. It empowers users to effortlessly manage account details, check music availability, and stay informed about borrowing limits. Tailored for both librarians and music enthusiasts, the system serves as a comprehensive user interface, streamlining library processes and interactions. Its versatility makes it applicable to any music library, facilitating efficient cataloging, borrowing, insertion, and monitoring activities. Developed in Java for superior performance, the system is a forward-looking solution adaptable to diverse operational contexts.

In parallel, the Music App enhances the music listening experience with a user-friendly interface, personalized playlists, and seamless streaming. Designed for enthusiasts and casual listeners, the app employs smart music discovery algorithms for dynamic recommendations. The offline mode adds flexibility, allowing users to download and access favorite tracks without an internet connection. Committed to innovation, the app continuously evolves, providing users with an engaging and up-to-date music exploration experience. Built on robust and scalable technologies it, ensures a user-friendly appeal.

### 1.4 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

### 1.5 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)

## 2. Overall Descriptions

### 2.1 Product Perspective

Use Case Diagram of Library Management System

This is a broad level diagram of the project showing a basic overview. The users can be either staff or student.. This System will provide a search functionality to facilitate the search of resources. This search will be based on various categories viz. book name or the ISBN. Further the library staff personnel can add/update the resources and the resource users from the system.The users of the system can request issue/renew/return of books for which they would have to follow certain criteria.

### 2.2 Product Function

Entity Relationship Diagram of Library Management System

The Online Library System provides online real time information about the books available in the Library and the user information. The main purpose of this project is to reduce the manual work. This software is capable of managing Book Issues, Returns, Calculating/Managing Fine, Generating various Reports for Record-Keeping according to end user requirements. The Librarian will act as the administrator to control members and manage books. The member’s status of issue/return is maintained in the library database. The member’s details can be fetched by the librarian from the database as and when required. The valid members are also allowed to view their account information.

### 2.3 User Classes and Characteristics

The system provides different types of services based on the type of users [Member/Librarian]. The Librarian will be acting as the controller and he will have all the privileges of an administrator. The member can be either a student or staff of the university who will be accessing the Library online.

The features ofAdmin are:

* Issue and return music items to members.
* View different categories of music available in the library.
* Access lists of music items available in each category.
* Handle the return of music items from members.
* Add new music items and their details to the database.
* Edit information for existing music items.
* Generate reports for existing music items and issued music.
* Access all accounts of the library members.

The features of Member:

* View different categories of music available in the library.
* Access lists of music items available in each category.
* Create and manage a personal account in the library.
* View the music items currently issued to them.
* Submit requests for new music items.
* Review the history of previously issued music items.
* Search for specific music titles in the library's collection.

**2.4 Operating Environment**

### This Music Library Management System (MLMS) is designed to operate seamlessly within the Windows environment. As a web application, it will be accessible through all major browsers, with specific optimization for Microsoft Internet Explorer, Google Chrome, and Mozilla Firefox.

### Compatibility:

### Full support:

### Microsoft Internet Explorer 6.0 and above

### Google Chrome

### Mozilla Firefox

### Limited compatibility:

### Mozilla Firefox & Opera 7.0 or higher versions (may require specific configuration for certain features)2.5 Assumptions and Dependencies

The assumptions are:-

* It is imperative that the coding be free of errors and that the system be straightforward to use for users.
* A database that is accessible through the website is required to hold the data of all users, books, and libraries.
* More storage space and quick database access should be features of the system.
* The system ought to offer a search function and facilitate speedy transactions.
* The Music System operates around the clock.
* Users can access from any computer with an Internet connection and the ability to browse the Internet.
* To access their online accounts and take action, users need to have the necessary usernames and passwords.

The dependencies are:-

* The specific hardware and software due to which the product will be run
* On the basis of listing requirements and specification the project will be developed and run
* The end users (admin) should have proper understanding of the product
* The system should have the general report stored
* The information of all the users must be stored in a database that is accessible by the Library System
* Any update regarding the book from the library is to be recorded to the database and the data entered should be correct

### 2.6 Requirement

Software Configuration:-

This software package is developed using java as front end . Microsoft SQL Server as the back end to store the database.

Operating System: Windows NT, windows 98, Windows XP

Language: Java Runtime Environment, IntelliJ IDEA (front end)

Database: MS SQL Server (back end)

Hardware Configuration:-

Processor: Pentium(R)Dual-core CPU

Hard Disk: 40GB

RAM: 256 MB or more

**File Support**: Ability to support various music file formats.

### 2.7 Data Requirement

A music library management system requires data such as song titles, artists, albums, genres, and release years. It also needs the file paths of the songs. User data like playlists, ratings, and play counts are also important. Metadata for each track, including duration and bit rate, can be useful. Album art and artist images can enhance the user interface. The system should be able to handle updates to this data as the music library grows and changes.

**3. External Interface Requirement**

### 3.1 GUI

The software provides a good graphical interface for both the user and the administrator to manage the music library.

* It allows users to view reports like songs played during a specific time.
* It provides a search facility based on different criteria like song title, artist, or album.
* The user interface is customizable by the administrator and all modules must fit into this graphical user interface.

Authentication Page:

Users can register and create an account. If the username or password is incorrect, an error message appears.

Search:

Users can search for songs by entering the song title, artist, or album.

Genre Display:

This shows the genres of music available and allows the administrator to add/edit or delete genres from the list.

Management Console:

This allows the administrator to add/remove users, add, edit, or remove songs, and manage user permissions.

## 4. System Features

## It should be made clear to users of the music library system that their accounts are safe. This can be accomplished by offering:

* User authentication and validation of members using their unique username or email address.
* Proper monitoring by the administrator which includes updating account status, showing a popup if the member attempts to stream a number of songs that exceed the limit provided by the music library policy, assigning penalties to members who violate the terms of service.
* Appropriate responsibility, which entails preventing a member from viewing the accounts of other members. All member accounts are visible and manageable only to the administrator.

## 5. Other Non-functional Requirements

### 5.1 Performance Requirement

The performance requirements for a Music Library Management System could be as follows:

Speed and Efficiency: The system should be fast and efficient in processing requests, such as searching for songs, streaming music, and managing user accounts. This includes quick response times for user queries and efficient use of system resources.

Error Handling: The system should be able to handle both expected and unexpected errors in a way that prevents loss of information and minimizes downtime. This includes built-in error testing to identify invalid usernames or passwords, and mechanisms to recover from errors without causing disruption to the user.

Scalability: The system should be able to handle a large amount of data and accommodate a high number of songs and users without any fault. This means the system should be designed to scale up as the number of users or the amount of data increases, without a significant impact on performance.

Reliability: The system should be reliable and available whenever users want to access it. This means ensuring high uptime, regular backups of data to prevent loss, and robust security measures to protect against unauthorized access.

User Experience: The system should provide a smooth and seamless user experience. This includes intuitive navigation, personalized recommendations, high-quality audio streaming, and easy-to-use controls for playing, pausing, and skipping tracks.

These performance requirements ensure that the Music Library Management System is robust, secure, and capable of delivering a high-quality service to its users.

### 5.2 Safety Requirement

The Music Library Management System should prioritize user data privacy, secure authentication, and robust error handling. It must protect against unauthorized access, ensuring a safe environment for users to enjoy music. Regular system audits and updates should be conducted to maintain the highest level of security.

### 5.3 Security Requirement

* Secure login systems to validate users.
* Encrypt sensitive user data for protection.
* Restrict unauthorized access to user accounts.
* Identify and handle system errors securely.
* Conduct security audits to identify vulnerabilities.

### 5.4 Requirement attributes

### The project may be created by more than one administrator, and each of them will have the authority to make system modifications. However, users or other members are unable to make modifications.

### It should be an open-source project.

### The quality of the database is preserved in a way that makes it extremely user-friendly for all users, making it simple for them to locate and play their favorite music.

### The database is kept up to date in a way that makes it easy for all users to search and play their favorite songs.

### Users can download and install the system with ease.

### 5.5 Business Rules

The Music Library Management App emphasizes copyright compliance, user activity monitoring, and content quality standards. It fosters a positive environment with clear community guidelines and data privacy rules. The app features collaborative playlists, a customizable notification system, and efficient feedback mechanisms. It ensures device compatibility and transparent communication about system updates. These rules enhance functionality, user experience, and address legal compliance, content quality, community interaction, and user privacy.

### 5.6 User Requirement

The Music Library Management App delineates distinct roles for its users, categorizing them into members with basic computer and internet skills and administrators possessing in-depth system knowledge. Members benefit from a user-friendly interface, comprehensive user manuals, and online help guides to navigate the system seamlessly. Administrators, adept at addressing technical issues, play a pivotal role in maintaining system integrity, promptly rectifying problems arising from disk crashes, power failures, and other potential catastrophes. The administrators provide crucial facilities like:

* Forgot Password feature,
* Data replication for backup,
* Auto-recovery mechanisms
* Efficient file organization.

These measures collectively create a user-centric environment, empowering members with accessible tools and providing administrators with the capabilities needed to uphold the system's robustness and resilience.

## 6. Other Requirements

### 6.1 Data and Category Requirement

The structure proposed for the Music Library Management App effectively acknowledges the diverse preferences of users, allowing them to customize playlists, add, delete, and modify songs across various genres, languages, and artists. Ensuring uniform access controls for all users fosters inclusivity and a seamless user experience. Additionally, the incorporation of different song categories based on genres, artists, and languages aligns with the varied tastes of the user base. However, to enhance clarity and user experience, it might be beneficial to specify the exact format in which categories and related data should be coded. This ensures consistency in data representation, making it easier for both users and administrators to navigate and manage the music library efficiently. Overall, the foundation is solid, and specifying the coding format will further refine the implementation of song categories.

### 6.2 Appendix

A: Admin, Abbreviation, Acronym,Artist, Assumptions,Album; B:Business rules; C: Class, Client, Conventions; D: Data requirement, Dependencies; G: GUI,Genre; K: Key; M: Music library,Member,Music Licensing,Metadata; N: Non-functional Requirement; O: Operating environment; P: Playlist,Performance,Perspective,Purpose; R: Requirement, Requirement attributes; S: Safety, Scope, Security, System features,Streaming; T:Track;U: User Profile, User class and characteristics, User requirement;

### 6.3 Glossary

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

* Administrator:An administrator is a user who has rights to manage users in thesoftware**.**
* User: A username 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
* Streaming: The process of playing music directly from the server without downloading it to the user’s device.
* Playlist: A user-curated collection of songs.
* Album: A collection of songs released by an artist or a band.
* Genre: A category of music (e.g., rock, jazz, classical).
* Artist: The individual or group who performs the music.

### 6.4 Class Diagram

A class diagram for a Music Library Management System provides a visual representation of the system’s structure, illustrating the system’s classes, their attributes, and the relationships between them. It serves as a blueprint for the system, detailing elements such as Users, Playlists, Albums, Tracks, and Genres. Each class encapsulates specific attributes and behaviours relevant to its role. For instance, the User class may include attributes like username and password, while the Playlist class might encompass a list of tracks. Relationships between classes, such as a User creating a Playlist or a Playlist containing Tracks, are also depicted. This diagram is crucial for understanding and developing the system’s object-oriented design.