Music Player

Software Requirements Specification

29-10-2023

Chirag Sangwan

Prepared for
Continuous Assessment 3
Autumn 2023

=

Table of Contents

1.Introduction

1.1 Purpose

The purpose of this document is to provide a comprehensive overview of the requirements for the development of a music player software application. This SRS outlines the functional and non-functional requirements that the software must fulfill to meet user expectations and business objectives.

1.2 Scope

The music player application described in this document is intended to be a versatile, user-friendly, and cross-platform solution for playing, managing, and organizing audio files. It aims to cater to a broad user base with diverse audio file formats and playback needs.

1.3 Definitions, Acronyms, and Abbreviations

- **SRS** - *Software Requirements Specification: A document that outlines the functional and non-functional requirements of a software system.*
- **API** - *Application Programming Interface: A set of rules and protocols that allow different software applications to communicate with each other.*
- **GUI** - *Graphical User Interface: The visual interface that allows users to interact with the software using graphical elements.*
- **FLAC** - *Free Lossless Audio Codec: A popular audio file format known for its lossless compression.*
- **MP3** - *MPEG Audio Layer III: A widely used audio file format that uses lossy compression.*
- **WAV** - *Waveform Audio File Format: A standard audio file format used for high-quality audio.*
- **UX** - *User Experience: The overall experience and satisfaction that users have while interacting with the software.*

- **RAM** - *Random Access Memory: The computer's volatile memory used for running software and storing data during execution.*
- **CPU** - *Central Processing Unit: The main processing unit of a computer that executes instructions.*
- **SSL** - *Secure Sockets Layer: A security protocol used to establish secure connections over a network.*
- **API Key** - *A unique identifier or token used to authenticate and authorize access to an API or service.*
- **Metadata** - *Data that describes other data, such as information about audio tracks (e.g., title, artist, album).*
- **EULA** - *End User License Agreement: A legal contract between the software provider and the user, defining the terms of software usage.*

1.4 References

- *ISO/IEC/IEEE 29148:2018 - Systems and Software Engineering - Life Cycle Processes - Requirements Engineering*
- *User feedback and surveys from previous versions of the music player*
- *Relevant industry standards and best practices*
- *Copyright and licensing documentation for third-party libraries or services used in the application.*

1.5 Overview

The remainder of this document will elaborate on the functional and non-functional requirements of the music player application. It will provide detailed information on user registration and authentication, audio playback, music library management, audio metadata handling, equalizer functionality, cross-platform compatibility, offline mode, as well as non-

functional aspects like performance, security, usability, reliability, compatibility, updates and maintenance, and legal and licensing considerations.

2.General Description

This section provides an overall description of the music player software.

2.1 Product Perspective

In this section, you can discuss the relationship of the music player with other systems and its place within a larger context. For example, if the music player integrates with other software or hardware, or if it's part of a broader ecosystem.

2.2 Product Functions

This section outlines the high-level functions and features of the music player. What can users do with the application? For instance, it plays audio, manages playlists, and offers an equalizer.

2.3 User Characteristics

Describe the typical users of the music player. What are their characteristics? This could include their technical expertise, age group, or specific user needs.

2.4 General Constraints

List any constraints that affect the development or use of the music player. This could include budgetary constraints, platform limitations, or other restrictions.

2.5 Assumptions and Dependencies

State any assumptions made during the requirements gathering process and dependencies that the project relies on. For example, you might assume that

users have access to an internet connection for downloading updates or depend on third-party libraries for specific functionality.

3. Specific Requirements

This section delves into specific details regarding external interfaces.

3.1 External Interface Requirements

3.1.1 User Interfaces

This section specifies the requirements related to the user interface of the music player. It can include details such as the layout, navigation, and the appearance of the user interface.

3.1.2 Hardware Interfaces

Describe the hardware interfaces that the music player interacts with. This could include details on how the application interacts with hardware components like speakers, headphones, and external audio devices.

3.1.3 Software Interfaces

Specify any software interfaces that the music player depends on or interacts with. For example, if it integrates with a music store or relies on specific libraries or APIs for audio decoding.

3.1.4 Communications Interfaces

Outline the communication interfaces that the music player uses. This might include network protocols if the application streams music from online sources or communicates with other devices.

3.2 Functional Requirements

This section outlines the specific functionalities and features of the music player.

3.2.1 Audio Playback

Specify the requirements related to audio playback. This includes functions like play, pause, stop, skip, rewind, volume control, and support for various audio file formats (e.g., MP3, WAV, FLAC).

3.2.2 Music Library Management

Detail the requirements for managing the music library, such as adding audio files, creating and managing playlists, searching and filtering music by artist, album, genre, and other metadata.

3.2.2 Audio Metadata

Specify how the music player handles audio metadata. This includes automatic retrieval and display of metadata such as song title, artist, album, and album art.

3.2.3 Equalizer

Define the requirements for the equalizer functionality, such as the ability for users to adjust audio settings for an enhanced listening experience.

3.2.4 User Registration and Authentication

Specify the requirements related to user registration and authentication, including user account creation, login, and password reset.

3.2.5 Cross-Platform Compatibility

Detail the requirements for ensuring the music player's compatibility with various platforms (e.g., Windows, macOS, Android, iOS).

3.2.6 Offline Mode

Define the requirements for offline functionality, such as the ability for users to download music for offline listening.

3.3 Non-Functional Requirements

This section outlines the non-functional requirements for the music player.

3.3.1 Performance

Specify performance-related requirements, including response times, latency, and resource utilization. For example, the music player should have low latency during audio playback and responsive user interactions.

3.3.2 Reliability

Define reliability requirements, such as the system's ability to operate without errors and handle exceptional situations gracefully. For instance, the music player should not crash when encountering errors.

3.3.3 Availability

Specify the availability requirements, which might include the percentage of uptime the music player should maintain and how it handles server outages (if applicable).

3.3.4 Security

Detail the security requirements, including how user data is stored, transmitted, and protected. Consider access control, encryption, and compliance with relevant security standards.

3.3.5 Maintainability

Define requirements related to the ease of maintaining and updating the software. This may include support for software updates, bug fixes, and changes to the application.

3.3.6 Portability

Outline the requirements for portability, specifying how the music player should be compatible with different platforms (e.g., operating systems) and devices.

.

3.3 Design Constraints

Design constraints refer to limitations and requirements that shape the design of a software system. In the context of a music player application, here are some potential design constraints you might need to consider.

Platform Constraints: *The music player may need to run on multiple platforms, such as Windows, macOS, Android, and iOS. Designing for cross-platform compatibility can be a constraint, as each platform may have its own requirements and limitations.*

Hardware Constraints: *The software should be designed to work within the limitations of the hardware it's running on, such as the amount of available memory (RAM), processor speed, and storage space.*

Network Constraints: *If the music player relies on online streaming or synchronization features, it should be designed to work within the constraints of various network conditions, including low bandwidth or intermittent connectivity.*

User Interface Guidelines: *Different platforms have their own user interface design guidelines. Your design must conform to these guidelines to ensure a consistent and user-friendly experience. For example, adhering to Material Design for Android apps or Human Interface Guidelines for iOS.*

Legal and Licensing Constraints: *The design should adhere to copyright and licensing laws, ensuring that the application doesn't enable or facilitate the illegal distribution of copyrighted material.*

Accessibility Requirements: *Design constraints may require the software to be accessible to people with disabilities. This includes designing for screen readers, keyboard navigation, and ensuring that the user interface is usable by individuals with visual or auditory impairments.*

Performance Constraints: *The design should consider the performance of the application. It needs to operate efficiently to provide smooth audio playback and responsiveness to user interactions, even on devices with lower hardware specifications.*

Security Constraints: *The design should implement security measures to protect user data and ensure secure communication, especially if the application involves user accounts, payment processing, or sensitive information.*

Third-Party Software and Libraries: *If the application relies on third-party software or libraries, design constraints might involve compatibility with specific versions of these components or adhering to their licensing terms.*

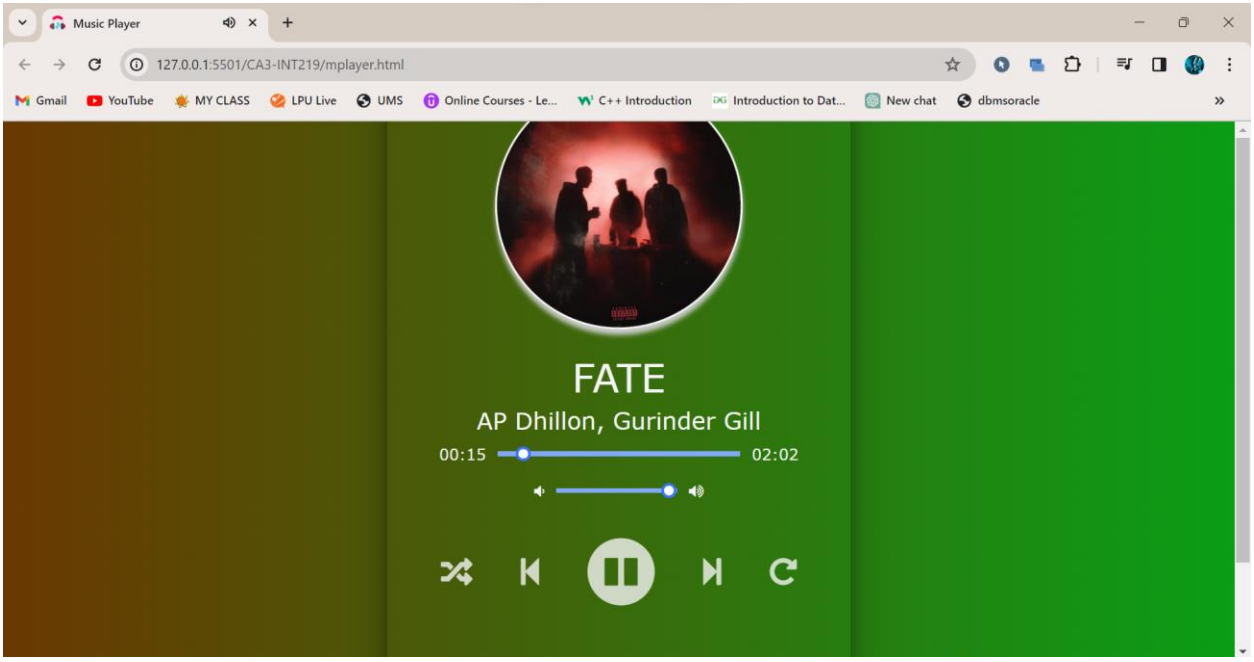
Localization Constraints: *If the music player is intended for a global audience, the design should consider support for multiple languages and localization of content.*

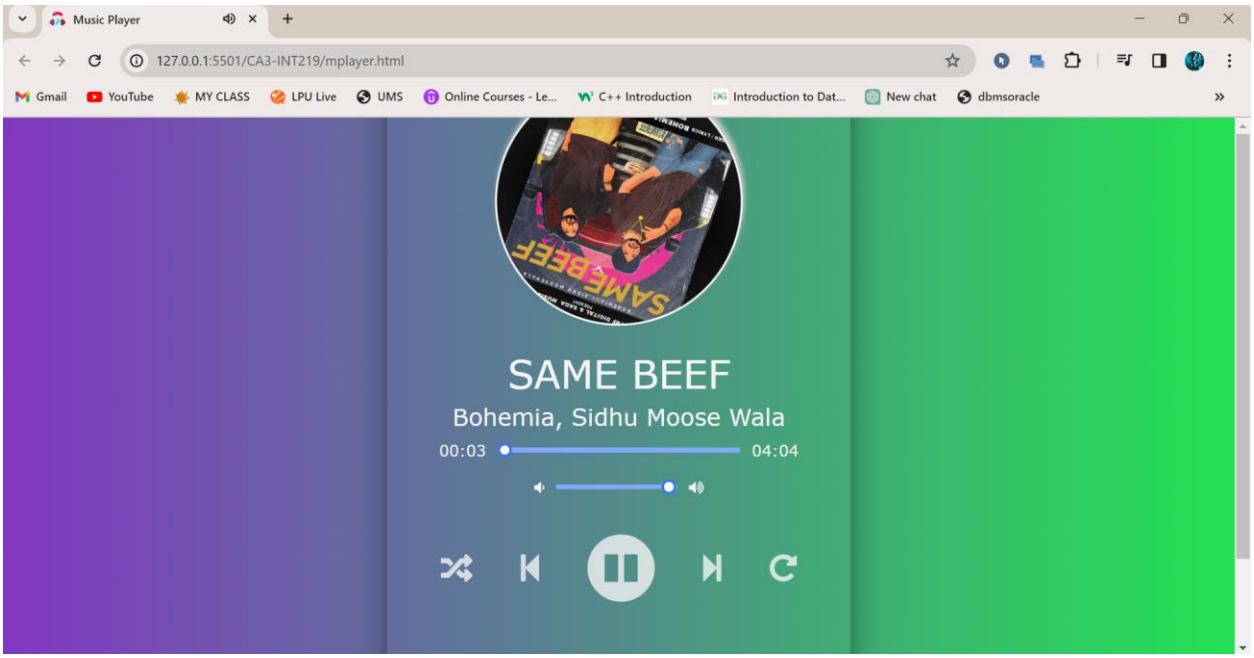
Scalability Constraints: *If the music player can interact with an online service, the design should account for scalability to handle many users and audio files.*

Regulatory Constraints: *Depending on the region or industry, there may be specific regulatory requirements that impact on the design. For example, compliance with data protection laws like GDPR.*

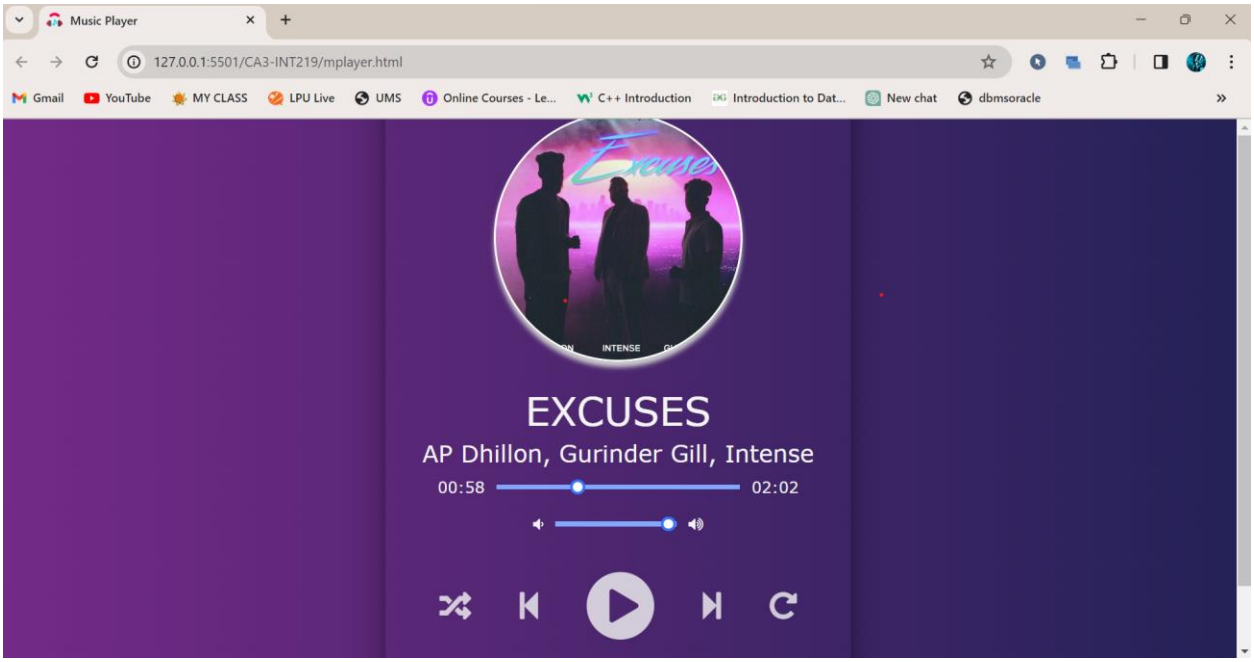
Budget and Resource Constraints: *The design may need to fit within a specific budget, which could limit the resources available for development.*

3.5 Snapshots of Project





Music Player



Software Requirements Specification