Software Requirements Specification

for

Melo-Track

Version 1.0

Prepared by: Michael Babatunde, Ty Hing, Megan Parnell

Farmingdale State College

February, 2024

Table of Contents

Table of Contents **ii**
Revision History **ii**
1. Introduction **1**
1.1 Purpose 1
1.2 Document Conventions 1
1.3 Intended Audience and Reading Suggestions 1
1.4 Product Scope 1
1.5 References 1
2. Overall Description **2**
2.1 Product Perspective 2
2.2 Product Functions 2
2.3 User Classes and Characteristics 2
2.4 Operating Environment 2
2.5 Design and Implementation Constraints 2
2.6 User Documentation 2
2.7 Assumptions and Dependencies 3
3. External Interface Requirements **3**
3.1 User Interfaces 3
3.2 Hardware Interfaces 3
3.3 Software Interfaces 3
3.4 Communications Interfaces 3
4. System Features **4**
4.1 System Feature 1 4
4.2 System Feature 2 (and so on) 4
5. Other Nonfunctional Requirements **4**
5.1 Performance Requirements 4
5.2 Safety Requirements 5
5.3 Security Requirements 5
5.4 Software Quality Attributes 5
5.5 Business Rules 5
6. Other Requirements **5**
Appendix A: Glossary **5**
Appendix B: Analysis Models **5**
Appendix C: To Be Determined List **6**

Revision History

Name	Date	Version
Melo-Track	02/28/24	1.0

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the MP3 Player Application. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, and the constraints under which it must operate.

1.2 Document Conventions

This document follows the IEEE standard for Software Requirements Specification documents.

1.3 Intended Audience and Reading Suggestions

This document is intended for developers, project managers, marketing staff, testers, and users of the MP3 Player Application. The section "Overall Description" gives a high-level view of the application, and the section "Specific Requirements" details the functionalities and system interfaces.

1.4 Product Scope

This software system will be an MP3 Player Application designed to manage and play MP3 files. The user can load MP3 files, create playlists, play songs, and control the playback (play, pause, stop, next, previous).

1.5 References

- *IEEE Standard for Software Requirements Specifications (IEEE 830-1998)*
- *MP3 Player User Manual*

2. Overall Description

2.1 Product Perspective

The MP3 Player Application is a standalone system that allows users to play MP3 files on their device.

2.2 Product Functions

- *Play, pause, stop, skip, and rewind functionality*
- *Playlist creation and management*

- *Equalizer settings*
- *Battery level indicator*
- *USB connectivity for file transfer*
- *Support for various audio formats (MP3, WAV, FLAC, etc.)*

2.3 User Classes and Characteristics

- *Casual Users: Individuals who use the MP3 player for leisure listening.*
- *Power Users: Individuals who require advanced features such as playlist management and equalizer settings.*
- *End Users: They will use the application to play and manage their MP3 files.*

2.4 Operating Environment

The MP3 Player is designed to operate in a wide range of environments, from indoor settings to outdoor settings. The application should be compatible with major operating systems like Windows, macOS, and Linux.

2.5 Design and Implementation Constraints

- *Device Compatibility: The MP3 player application must be compatible with a wide range of devices, including smartphones, tablets, and computers, running various operating systems such as iOS, Android, and Windows.*
- *Audio Formats: The application should support multiple audio formats commonly used for digital music files, including MP3, AAC, FLAC, WAV, and OGG, to ensure compatibility with a diverse library of music.*
- *Screen Sizes and Resolutions: The user interface (UI) design must be adaptable to different screen sizes and resolutions to provide a consistent and intuitive user experience across various devices, including smartphones, tablets, and desktop computers.*
- *Memory and Storage Limitations: The application should be optimized to minimize memory usage and efficiently manage storage space, especially on devices with limited RAM and internal storage capacity, to prevent performance issues and ensure smooth operation.*

2.6 User Documentation

User documentation will be provided in the form of a user manual and online help. Introduction, Getting Started, Playing Music, Creating Playlists, Navigation, Settings

2.7 Assumptions and Dependencies

Operating System Dependency: The MP3 player application assumes compatibility with specific operating systems, such as Android, iOS, or Windows, to run on smartphones, tablets, or computers.

Hardware Compatibility: The application depends on the availability of compatible hardware components, including audio output devices (e.g., headphone jack, Bluetooth), storage media (e.g., internal memory, microSD card), and user input interfaces (e.g., physical buttons, touch screen).
Audio File Formats: The application assumes support for standard audio file formats, such as MP3, AAC, WAV, and FLAC, to play music files stored on the device.

3. External Interface Requirements

3.1 User Interfaces

The MP3 Player shall have a display screen showing the current track, playlist queue and other relevant information. It shall have buttons for play, pause, stop, skip, and volume control.
Main screen: Displays the list of songs and controls for playback.
Playlist screen: Allows the user to create and manage playlists.

3.2 Hardware Interfaces

The application will interface with the device's file system to load MP3 files and the device's sound hardware to play them.
Audio Output Interface:

- *Purpose: Connects the MP3 player application to audio output devices for sound playback.*
- *Examples: Headphone jack, Bluetooth connectivity for wireless headphones or speakers, auxiliary output.*

User Input Interface:

- *Purpose: Allows users to interact with the MP3 player application and control its functions.*
- *Examples: Physical buttons for playback control (play, pause, skip), touch screen interface for navigation and selection.*

3.3 Software Interfaces

User Interface (UI)

- *Purpose: Allows users to interact with the MP3 player application.*
- *Features: Display of music library, playback controls (play, pause, skip), volume control, playlist management.*
- *Compatibility: Designed for intuitive use on various devices (smartphones, tablets, computers).*

File Management Interface

- *Purpose: Manages the organization and storage of music files on the device.*
- *Features: File browsing, folder navigation, file transfer (uploading, downloading).*
- *Compatibility: Supports various file formats (MP3, AAC, FLAC) and file management operations.*

3.4 Communications Interfaces

Bluetooth Interface

- *Purpose: Enables wireless connectivity with compatible devices such as headphones, speakers, and smartphones.*
- *Features: Wireless audio streaming, control playback remotely.*
- *Compatibility: Compatible with Bluetooth-enabled devices.*

4. System Features.>

- User authentication and secure data storage
- Responsive application with playback controls to ensure audio quality
- Interactive tools for commenting/liking friends' playlists
- Storage capacity to allow users to build their playlists

4.1 User authentication and secure data storage

4.1.1 Description and Priority

This feature ensures that users can securely create and access their accounts, protecting their personal information and playlists. It involves implementing robust authentication mechanisms and encrypting user data both at rest and in transit. This feature is of high priority due to its critical role in user trust and legal compliance. The priority components could be rated as follows: Benefit (9), Penalty (9), Cost (5), Risk (8). The high benefit and penalty scores reflect the importance of security for user retention and legal compliance, while the cost and risk scores acknowledge the investment and potential challenges in implementing state-of-the-art security measures.

4.1.2 Stimulus/Response Sequences

1. User Sign-Up: The user provides their email and password. The system validates the input, creates the account, encrypts the password, and stores it securely.
2. User Login: The user enters their email and password. The system verifies the credentials and grants access if they match.
3. Password Reset: The user requests a password reset. The system sends a secure link to their email. The user clicks the link and enters a new password, which is then encrypted and stored.
4. Data Access: The user accesses their playlists. The system retrieves the data, decrypting it on-the-fly for display.
5. User Logout: The user logs out. The system ends the session and clears any temporary authentication tokens.

4.1.3 Functional Requirements

REQ-1: Authentication System - The system must support secure user authentication, including password encryption and session management. It should prevent unauthorized access and support multi-factor authentication (MFA) for enhanced security.

REQ-2: Data Encryption - All personal and playlist data stored in the system must be encrypted using industry-standard encryption algorithms. This includes both data at rest and data in transit.

REQ-3: Password Management - The system must provide a secure mechanism for password reset and recovery, including email verification and temporary secure links for password reset procedures.

REQ-4: Access Control - The system must implement role-based access control (RBAC) to ensure users can only access their data and relevant functionalities.

REQ-5: Audit Logging - The system should log all access and modification actions to user data and authentication attempts, to support security audits and troubleshooting.

REQ-6: Data Backup and Recovery - Implement regular data backups and a secure recovery process to prevent data loss and ensure continuity in case of system failures.

REQ-7: Secure Data Deletion - When a user deletes their account or specific data, the system must securely erase the data to prevent recovery.

REQ-8: Error Handling and Notification - The system must gracefully handle errors and invalid inputs, providing clear error messages to users. It should also notify users of suspicious activities related to their account.

4.2 Responsive Application with Playback Controls to Ensure Audio Quality

4.2.1 Description and Priority

This feature aims to offer users a seamless and high-quality audio playback experience across various devices. It includes responsive design elements that adapt to different screen sizes and resolutions, along with intuitive playback controls such as play, pause, skip, rewind, and volume adjustment. Ensuring high audio quality regardless of the user's device or network conditions is also a critical aspect. This feature is of high priority because it directly affects user satisfaction and engagement. The priority components could be rated as follows: Benefit (9), Penalty (8), Cost (4), Risk (5). High benefit and penalty scores reflect the feature's importance in user experience and retention, while cost and risk are moderately rated, acknowledging the technical challenges and the need for thorough testing across devices.

4.2.2 Stimulus/Response Sequences

1. Screen Size Adaptation: The user accesses the application on a device. The system detects the screen size and resolution, adjusting the layout and controls for optimal usability.
2. Playback Control Interaction: The user clicks the play button. The system responds by playing the selected audio track and displaying an updated playback status.
3. Audio Quality Adjustment: The system automatically adjusts the audio stream quality based on the user's current network speed to ensure continuous playback without buffering.
4. Manual Quality Selection: The user selects a preferred audio quality setting. The system applies this setting for future playback, overriding automatic adjustments.
5. Error Handling: In case of a playback error (e.g., file not found or network issue), the system displays an informative error message and suggests actions to the user (e.g., try again, check network connection).

4.2.3 Functional Requirements

REQ-1: Responsive Design - The application must automatically adjust its layout and playback controls to fit the screen size and resolution of the user's device, ensuring usability and accessibility.

REQ-2: Playback Controls - Implement intuitive playback controls, including play, pause, skip, rewind, and volume adjustment, allowing users to easily navigate their audio content.

REQ-3: Audio Quality Management - The system must automatically adjust the audio quality based on the user's network conditions to ensure smooth playback without interruptions.

REQ-4: Manual Quality Selection - Users should be able to manually select their preferred audio quality from available options, which the system will prioritize for future playback.

REQ-5: Error Handling and Feedback - The application must provide clear feedback in case of playback issues or errors, including actionable suggestions for the user to resolve the issue.

REQ-6: Device Compatibility Testing - Conduct extensive testing across a wide range of devices and operating systems to ensure consistent functionality and performance of the playback controls and responsive design.

REQ-7: User Preferences Storage - The system should store user preferences, including manual quality settings and other playback options, and apply these settings automatically during future sessions.

REQ-8: Adaptive Streaming - Implement adaptive streaming technologies to dynamically adjust audio streams to the optimal quality level based on real-time network conditions.

4.3 Interactive Tools for Commenting/Liking Friends' Playlists

4.3.1 Description and Priority

This feature introduces social interaction capabilities, enabling users to engage with their friends' playlists through comments and likes. It aims to foster a community atmosphere, encouraging users to share opinions, recommendations, and feedback directly on playlist pages. Given its role in enhancing user engagement and retention by leveraging social connections, this feature is considered to have a High priority. Priority components might be rated as follows: Benefit (8), Penalty (7), Cost (5), Risk (4). The high benefit reflects the positive impact on user engagement and social interaction within the app, while the moderate cost and risk acknowledge the need for implementing moderation tools and privacy settings.

4.3.2 Stimulus/Response Sequences

1. Liking a Playlist: A user clicks the like button on a friend's playlist. The system increments the like count and displays the updated total to all viewers.
2. Commenting on a Playlist: A user submits a comment on a friend's playlist. The system posts the comment in real-time, visible to all friends or users with access.
3. Notification of Interaction: The owner of the playlist receives a notification about the new like or comment, promoting further engagement.
4. Privacy Settings Adjustment: A user adjusts their playlist's privacy settings. The system updates who can comment on or like the playlist accordingly.
5. Moderation of Comments: A user reports an inappropriate comment. The system flags the comment for review by moderators and temporarily hides it from public view.

4.3.3 Functional Requirements

REQ-1: Commenting Functionality - Implement a feature allowing users to post comments on friends’ playlists, including support for text input and immediate display upon submission.

REQ-2: Like System - Provide a like button for each playlist that users can click to express their enjoyment, with a counter displaying the total number of likes.

REQ-3: Real-time Notifications - Implement a notification system to alert playlist owners about new comments and likes, encouraging active participation and engagement.

REQ-4: Privacy Controls - Allow users to set privacy levels for their playlists, determining who can comment on or like their playlists (e.g., everyone, friends only, no one).

REQ-5: Comment Moderation - Include tools for users to report inappropriate comments and for moderators to review and remove them, ensuring a positive and respectful community environment.

REQ-6: User Feedback Loop - Provide users with the ability to delete their own comments or likes and to see who has liked their playlists, fostering transparency and control.

REQ-7: Integration with User Profiles - Ensure that interactions (likes and comments) are integrated with user profiles, allowing others to explore playlists through social connections.

REQ-8: Scalable Architecture - Design the commenting and liking system to be scalable, supporting a large number of concurrent users and interactions without performance degradation.

4.4 Storage Capacity to Allow Users to Build Their Playlists

4.4.1 Description and Priority

This feature ensures that users have ample storage capacity to build and save their personalized music playlists. It includes functionalities for creating, editing, organizing, and storing playlists with ease. By allowing users to curate their music, this feature significantly enhances user satisfaction and engagement with the app. It is of **High priority** due to its fundamental role in user experience and content personalization. The priority components could be rated as follows: Benefit (9), Penalty (8), Cost (6), Risk (3). High benefit and penalty scores reflect the critical importance of personalization in user retention and satisfaction, while the cost and risk scores recognize the investment in storage infrastructure and the relatively low technical risk.

4.4.2 Stimulus/Response Sequences
1. Creating a Playlist: A user selects the option to create a new playlist, inputs a name, and adds songs. The system saves the playlist to the user's account.
2. Editing a Playlist: A user adds or removes songs from an existing playlist. The system updates the playlist content in real-time.
3. Accessing Playlists: A user navigates to their playlist library. The system displays all saved playlists, allowing the user to select and play any playlist.
4. Organizing Playlists: A user rearranges the order of playlists or songs within a playlist. The system saves these changes immediately.
5. Deleting a Playlist: A user decides to delete a playlist. The system prompts for confirmation and, upon user agreement, permanently removes the playlist.

4.4.3 Functional Requirements
REQ-1: Playlist Creation and Management - The system must allow users to create new playlists, add or remove songs, and edit playlist details (e.g., name, description).

REQ-2: Ample Storage Capacity - Ensure sufficient storage capacity for users to save a large number of playlists and songs, accommodating diverse user needs and growing libraries.

REQ-3: User Interface for Playlist Organization - Provide an intuitive user interface for organizing playlists, including drag-and-drop functionality for rearranging songs and playlists.

REQ-4: Playlist Access and Playback - Users should be able to easily access their playlists from any device, with support for seamless playback and the ability to shuffle or repeat songs.

REQ-5: Data Integrity and Backup - Implement measures to ensure the integrity of playlist data, including regular backups to prevent data loss and facilitate recovery if needed.

REQ-6: Performance and Scalability - The storage and playlist management system must be designed for high performance and scalability, ensuring quick access and modification of playlists as user data grows.

REQ-7: Security and Privacy - Protect user-created playlists with appropriate security measures, ensuring that only authorized users can access, edit, or delete them.

REQ-8: Cross-platform Synchronization - Ensure that playlists are synchronized across all devices where the user is logged in, providing a consistent experience regardless of the access point.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- The application must support real-time interactions, such as commenting and liking playlists, with a response time not exceeding 2 seconds under typical usage conditions.
- The system should handle simultaneous login requests for up to 10,000 users per minute, ensuring user authentication processes are not slowed during peak times.
- Playback controls should respond to user input without any noticeable delay, aiming for a maximum response time of 500 milliseconds to ensure a seamless audio experience.

5.2 Safety Requirements

- The product shall implement measures to prevent hearing damage, including limiting the maximum volume output and providing warnings for prolonged listening at high volumes.
- Adhere to data protection regulations such as GDPR or CCPA to safeguard user data against loss or misuse.

5.3 Security Requirements

- All user data, including authentication credentials and playlist information, must be encrypted both at rest and in transit using industry-standard encryption protocols (e.g., TLS, AES).
- Implement multi-factor authentication (MFA) for user login processes to enhance account security.
- Comply with relevant security certifications and standards, such as ISO/IEC 27001, to ensure the protection of user data.

5.4 Software Quality Attributes

- Adaptability: The system must be designed to easily incorporate additional social features or support new audio formats without significant overhauls.
- Availability: Ensure 99.9% uptime for the application, with redundancy and failover mechanisms in place to minimize disruptions.
- Maintainability: Code should be well-documented and modular to facilitate easy updates and maintenance.
- Usability: Prioritize intuitive design for the user interface to make navigation, playlist management, and social interactions straightforward for users of all technical skill levels.

5.5 Business Rules

- Only registered users can create playlists, comment on, or like playlists. This enforces a level of accountability and community moderation.
- Users must be friends or follow each other to interact with each other's playlists, promoting a sense of community and safety.

- Playlists marked as private by their creators cannot be commented on or liked by others, respecting user privacy and content control.

6. Other Requirements

Database Requirements
- The system must use a scalable, secure database capable of storing millions of user profiles, playlists, and track metadata efficiently.
- Database schemas should be designed to support complex queries for user interactions, playlist recommendations, and social features with minimal latency.

Internationalization Requirements
- The application must support multiple languages and regional settings, including but not limited to English, Spanish, French, and Mandarin. This includes user interface elements, messages, and date formats.
- Provide a framework for easy addition of new languages to accommodate expanding global user base.

Legal Requirements
- Comply with digital copyright laws and music licensing agreements in all operational regions.
- Adhere to data protection and privacy laws such as GDPR in Europe and CCPA in California, ensuring user data is handled legally and ethically.

Reuse Objectives
- Components of the application, such as the authentication system and playback controls, should be designed for reuse in other projects within the organization.
- Leverage open standards and interfaces to facilitate integration with third-party services and plugins.

Appendix A: Glossary

- API (Application Programming Interface): A set of protocols and tools for building software applications, specifying how software components should interact.
- GDPR (General Data Protection Regulation): A regulation in EU law on data protection and privacy in the European Union and the European Economic Area.
- CCPA (California Consumer Privacy Act): A state statute intended to enhance privacy rights and consumer protection for residents of California, USA.
- SRS (Software Requirements Specification): A document that describes what the software will do and how it will be expected to perform.
- TLS (Transport Layer Security): A cryptographic protocol designed to provide communications security over a computer network.
- AES (Advanced Encryption Standard): A symmetric encryption algorithm used worldwide to secure data.

Appendix B: Analysis Models

- Data Flow Diagrams (DFDs): Illustrate how data moves through the system, from user interactions to backend processes and database transactions.
- Class Diagrams: Provide an overview of the system's structure, showing classes, their attributes, methods, and the relationships between classes.
- State-Transition Diagrams: Describe how the application transitions between states, particularly in response to user interactions like playing or pausing music.
- Entity-Relationship Diagrams (ERDs): Map out the database schema, showing how entities such as users, playlists, and tracks relate to each other.

Appendix C: To Be Determined List

1. Final selection of database technology and schema design.
2. Specific third-party services for internationalization support.
3. Detailed compliance plan for GDPR and CCPA, including data handling, storage, and user consent mechanisms.
4. Complete list of languages and regions for initial release and internationalization strategy.
5. Decision on open standards and interfaces for component reuse and third-party integration.