Software Requirements Specification

1. PROBLEM DEFINITION

General Overview of the Project

MuseBoxd is a web application designed to revolutionize the way musicians, producers, DJs, and music enthusiasts engage with music discovery and networking. By offering tools for artists to build personal brands and connect with their audience while providing users with personalized music recommendations and community engagement, MuseBoxd bridges the gap between creators and listeners.

Problem Statement

Musicians and producers face challenges in gaining visibility and connecting with their audience. Simultaneously, music enthusiasts struggle to discover personalized music recommendations and engage meaningfully with the music community. Existing platforms often lack a unified experience for both artists and listeners. MuseBoxd addresses these gaps by creating a centralized, user-friendly platform that caters to the needs of both artists and music lovers.

Key Objectives

- Provide artists with tools for showcasing their work, managing tours, and generating revenue.
- Offer users personalized music discovery options, playlist creation, and opportunities to engage with artists.
- Foster community interaction through reviews, forums, and live music experiences.
- Ensure scalability, accessibility, and high performance to support a growing user base.

Components of the System

1. **Artist Features**: Profile creation, music upload, analytics, and tour management.
2. **User Features**: Music discovery, playlist management, reviews, and social interactions.
3. **Admin Features**: Content moderation, analytics, and control over featured content.
4. **APIs**: Integration with Spotify, YouTube, and Ticketmaster.
5. **Technology Stack**: React.js (frontend), Node.js with Express (backend), PostgreSQL (database), Cloud or Web (deployment).

Expected Benefits

- Enhanced visibility and revenue opportunities for artists.
- Improved music discovery and engagement for users.
- A vibrant community promoting global music exploration and appreciation.
- Scalable and secure platform adhering to modern standards.

2. REQUIREMENTS SPECIFICATION

Sources of Requirements

- **Artists**: Direct feedback about tools needed for music promotion and revenue tracking.
- **Users**: Insights on personalized music discovery and social features.
- **Admins**: Needs for moderation, analytics, and content management.
- **Market Research**: Analysis of competitors like Letterboxd and Spotify.
- **Regulations**: Compliance with GDPR and accessibility standards.

3. FUNCTIONAL REQUIREMENTS

Req. No	Requirement	Specific Example
FR01	Users can register and log in via SSO (Google, Facebook, etc.).	A user logs in using their Google account to access the platform.
FR02	Artists can upload, edit, and manage their discography.	A producer uploads an album and edits its details later.
FR03	Users can search and filter music by genre, artist, or release type.	A fan searches for rock music released in the last year.
FR04	Users can create, modify, and delete playlists.	A user creates a playlist for workout music and shares it on Twitter.

FR05	Artists can manage tour schedules and sell tickets via the platform.	A DJ updates their tour schedule and sells tickets through Ticketmaster integration.
FR06	Admins can moderate user-generated content.	Admins review and approve a user's review of an album.

4. SYSTEM REQUIREMENTS

- **Hardware Requirements:**
  - OS: Windows 10 or above, MacOS Mojave 10.14 or above.
  - Memory: 4GB or above.
  - Disk Space: 125GB or above.
  - Architecture: x32, arm, x64,x86.
  - Processor: Core 2 Duo or above.
- **Software Requirements:**
  - Operating System: Linux-based server for deployment.
  - Frontend: React.js framework.
  - Backend: Node.js with Express.
  - Database: PostgreSQL.
  - Deployment: Web Deployment.

5. CONCEPTUAL MODEL

Overview

The conceptual model of MuseBoxd represents the high-level interactions between its key components:

1. **Frontend**:
    - UI components for artist profiles, music search, playlists, and reviews.
    - Accessible via mobile and desktop browsers.

2. **Backend**:
    - RESTful API handling requests for user and artist data.
    - Middleware for authentication, data validation, and API integrations.

3. **Database**:
    - Tables for users, artists, music metadata, playlists, reviews, and analytics.

4. **APIs**:
    - Spotify for streaming music data.
    - Ticketmaster for event management and ticket sales.
    - YouTube for music video integration.

5. **Admin Tools**:
    - Dashboard for moderating content and managing featured artists.
    - Analytics for tracking platform usage and user behavior.

MuseBoxd's design focuses on addressing current challenges in music discovery and networking while contributing to a sustainable and engaging digital environment for artists and listeners alike.