Software Requirements Specification

for

DROPMUSE

Version 1.0

Prepared by

Aviskar Poudel [ACE079BCT013]

Instructor: Er. Ramesh Sharma

Course: Software Engineering [CT601]

Lab Section: A1

Date: July 20, 2025

Contents

Revision History

Version	Primary Author(s)	Description of Version	Date Completed
Draft 0.1	Aviskar Poudel	Initial draft version with project overview and basic requirements.	07/16/2025
Draft 0.2	Aviskar Poudel	Added functional requirements and updated system models.	07/18/2025
Draft 1.0	Aviskar Poudel	Added System block diagram, Mockup UI and Finalized SRS	07/19/2025

Table 1: Revision History

1 INTRODUCTION

1.1 Document Purpose

Managing and distributing music in today's digital age can be challenging for independent artists seeking to reach a global audience. Traditional distribution methods often involve complex procedures and limited transparency. **DropMuse** is a user-friendly, web-based music distribution platform that simplifies this process by allowing artists and labels to upload, manage, and distribute their music seamlessly to major global streaming platforms such as Spotify, Apple Music, and YouTube Music. The platform's intuitive interface, transparent revenue sharing, and integrated marketing tools empower artists to take full control of their music careers.

In addition to its global reach, **DropMuse** offers localized support tailored to the unique needs of artists from different regions, providing personalized assistance and resources to help them successfully navigate the digital music industry. By combining efficient distribution with actionable insights, community features, and affordable pricing, **DropMuse** stands out as a comprehensive solution designed to help independent musicians grow their audience, increase revenue, and build sustainable careers worldwide.

1.2 Product Scope

DropMuse creates a centralized platform for independent musicians, artists,and labels to upload, manage, and distribute their music to multiple global streaming services such as Spotify, Apple Music, and YouTube Music. Artists can create personal profiles, upload tracks along with metadata, and track the status of their releases. The platform also provides detailed analytics, revenue tracking, and marketing tools to help artists grow their audience and maximize their earnings. **DropMuse** ensures seamless management of all music distribution activities through an easy-to-use interface designed for artists at all experience levels.

Administrators manage artist and label accounts, review music submissions to maintain quality and compliance, and oversee the distribution workflow. **DropMuse** also offers localized support services, providing personalized assistance and resources tailored to artists' unique needs across different regions. By combining distribution, marketing, and support features in one platform, **DropMuse** aims to simplify the global music distribution process, helping artists expand their reach, increase revenue, and build sustainable careers in the music industry.

1.3 Intended Audience and Document Overview

The intended audience of this document primarily includes the lecturer who will evaluate the SRS as part of the lab assignment. Additionally, the document serves as a reference for developers who would implement the system, testers who would verify its functionality, and other stakeholders involved in the project. This ensures that all parties have a clear understanding of the system requirements to facilitate effective development and testing.

This SRS provides detailed descriptions of the internal and external system requirements, covering both functional and non-functional aspects of **DropMuse**. The document is organized to first present an overview of the product, including its purpose, scope, and key features, followed by detailed functional requirements, system constraints, and performance criteria. Readers should start with the overview and requirements sections to gain a complete understanding of the system.

1.4 Definitions, Acronyms and Abbreviations

- **API (Application Programming Interface)**: A set of definitions and protocols for building and integrating application software.
- **Artist**: A user who uploads, manages, and distributes their music content on the DropMuse platform.
- **AWS (Amazon Web Services)**: A secure cloud platform that provides computing power, storage, and other functionalities.
- **Client**: Refers to the end-user system or device accessing the DropMuse service.
- **CSS (Cascading Style Sheets)**: A stylesheet language used for describing the presentation and visual design of web pages written in HTML.
- **DBMS (Database Management System)**: Software that interacts with the user, applications, and the database to capture and analyze data.
- **DRM (Digital Rights Management)**: Technology used to protect copyrighted digital content from unauthorized use.

- **FLAC (Free Lossless Audio Codec)**: An audio format for lossless compression of digital audio, allowing original sound to be preserved exactly.
- **GUI (Graphical User Interface)**: A visual interface allowing users to interact with the system using graphical elements.
- **HTML (HyperText Markup Language)**: The standard language for creating and structuring content on the web.
- **ISRC (International Standard Recording Code)**: A unique identifier for sound recordings and music video recordings. Used to track and manage rights across digital platforms.
- **Label**: A verified entity that manages and distributes music from multiple artists.
- **MongoDB**: A NoSQL, document-oriented database used for storing structured and semi-structured data in JSON-like documents.
- **MP3 (MPEG-1 Audio Layer III)**: A standard digital audio encoding format that uses lossy compression to reduce file size while preserving sound quality for most applications.
- **OAuth 2.0**: An authorization framework that allows applications to obtain limited access to user accounts.
- **RESTful API (Representational State Transfer Application Programming Interface)**: An Application Programming Interface that conforms to the constraints of the REST architectural style and allows interaction with web services using standard HTTP methods.
- **SQL (Structured Query Language)**: A standardized language used to manage and manipulate relational databases. It is used for querying, inserting, updating, and deleting data stored in structured tables.
- **SRS (Software Requirements Specification)**: A detailed description of the system's intended capabilities, features, and constraints.
- **UI/UX (User Interface / User Experience)**: Design aspects that ensure ease of use and positive interaction between the user and the platform.
- **UML (Unified Modeling Language)**: A standardized modeling language used to visualize the design of a software system.
- **WAV (Waveform Audio File Format)**: An uncompressed audio format developed by Microsoft and IBM for storing high-quality audio.

1.5 Document Conventions

This Software Requirements Specification (SRS) document adheres to standard IEEE formatting guidelines, with adjustments to accommodate LaTeX styling. The document is typeset using the `Times` font. All section and subsection titles follow a hierarchical numbering format consistent with IEEE recommendations.

The main text is presented in a 12-point font size with single line spacing and 1-inch margins. Important terms and system components are emphasized using **boldface**. All acronyms and abbreviations are defined upon first usage and are listed in Section **1.4: Acronyms and Abbreviations** for reference.

1.6 References and Acknowledgments

1. IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications, IEEE Computer Society, 1998.
2. Sommerville, Ian. *Software Engineering*, 10th Edition, Pearson Education, 2015.
3. S. M. Fröschle, K. Pohl, "COMET: An Object-Oriented Method for Designing Concurrent Systems," *IEEE Transactions on Software Engineering*, 1996.
4. Object Management Group, "Unified Modeling Language (UML) Specification," Available at: `https://www.omg.org/spec/UML/`
5. OAuth 2.0 Authorization Framework, Internet Engineering Task Force (IETF), Available at: `https://oauth.net/2/`
6. Amazon Web Services (AWS) Documentation, Available at: `https://aws.amazon.com/documentation/`
7. Senior Design Project Resources, SRS Template, Iowa State University. Available at: `https://seniord.cs.iastate.edu/2020-May-02/files/inline-files/SRS%20Template.docx.pdf`
8. draw.io – Free Online Diagram Software. Available at: `https://www.draw.io`
9. Government of Nepal, Electronic Transactions Act, 2063 (2008). Available at: `https://www.lawcommission.gov.np/en/archives/category/documents/prevailing-law/statutes-acts/electronic-transactions-act-2063-2008`

10. Government of Nepal, Copyright Act, 2059 (2002). Available at: `https://www.lawcommission.gov.np/en/archives/2053`

Acknowledgments:
I would like to express my sincere gratitude to our course instructor, Er. Ramesh Sharma, for their invaluable guidance during the preparation of this Software Requirements Specification. I also acknowledge the support of open-source tools and platforms, including LaTeX and draw.io, that helped us in creating diagrams and formatting this document.

2 OVERALL DESCRIPTION

2.1 Product Overview

DropMuse is a new, standalone web-based platform designed to help the process of music distribution for independent artists and labels. Unlike traditional distribution channels that often involve multiple complex workflows and submission processes, **DropMuse** offers a centralized interface that allows users to upload their music, manage their artist profile, and distribute tracks to various global music platforms such as Spotify, Apple Music, YouTube Music, and others all from a single dashboard.

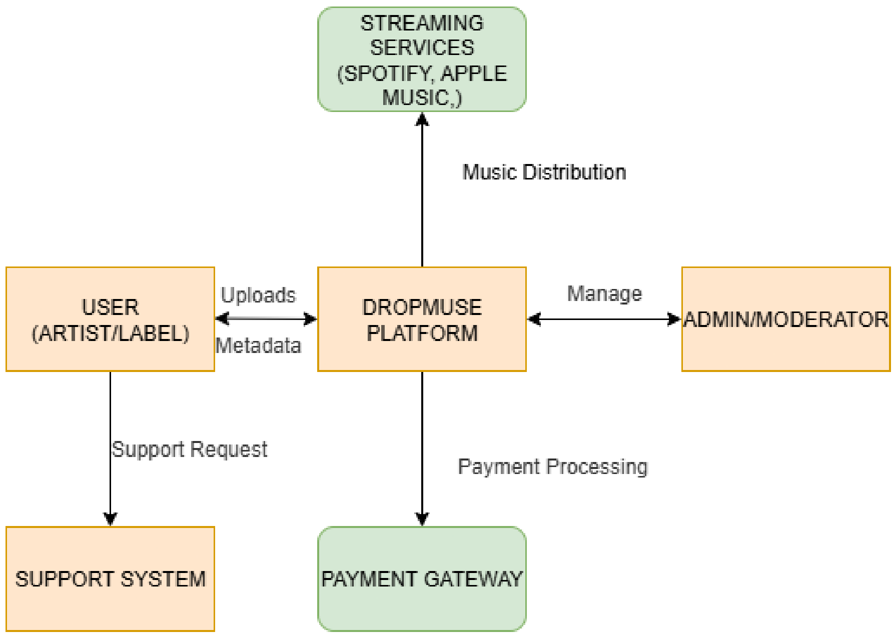

Figure 1: System Context Diagram of DropMuse

2.1.1 Concept of Operations

The main objective of **DropMuse** is to facilitate with a music distribution platform which is efficient and transparent. Artists or Labels access the platform through a web interface to upload their music tracks, input metadata, and initiate distribution to supported streaming platforms. The platform also offers administrative tools to review submissions, track analytics, and manage artist accounts. **DropMuse** gives features not only to independent artist but also labels, which can manage multiple artists and distribute their tracks through it.

2.1.2 Major User Interface

- **User Dashboard:** Allows musicians to upload tracks, manage their catalog, view distribution status, and analyze performance metrics.
- **Admin Panel:** Enables platform administrators to monitor content, approve submissions, and handle user support.

2.1.3 Hardware Interface

DropMuse supports any device capable of running modern web browsers with HTML5, CSS3, and JavaScript enabled, including desktops, laptops, tablets, and smartphones.

2.1.4 Software Interface

The platform relies heavily on JavaScript for frontend interactivity and API communication. It interfaces with external streaming services using their APIs to automate music publishing and metadata synchronization. The automation is not fully as there are admins and moderators who validate the rules and regulations of each music submission by user, for each streaming platforms.

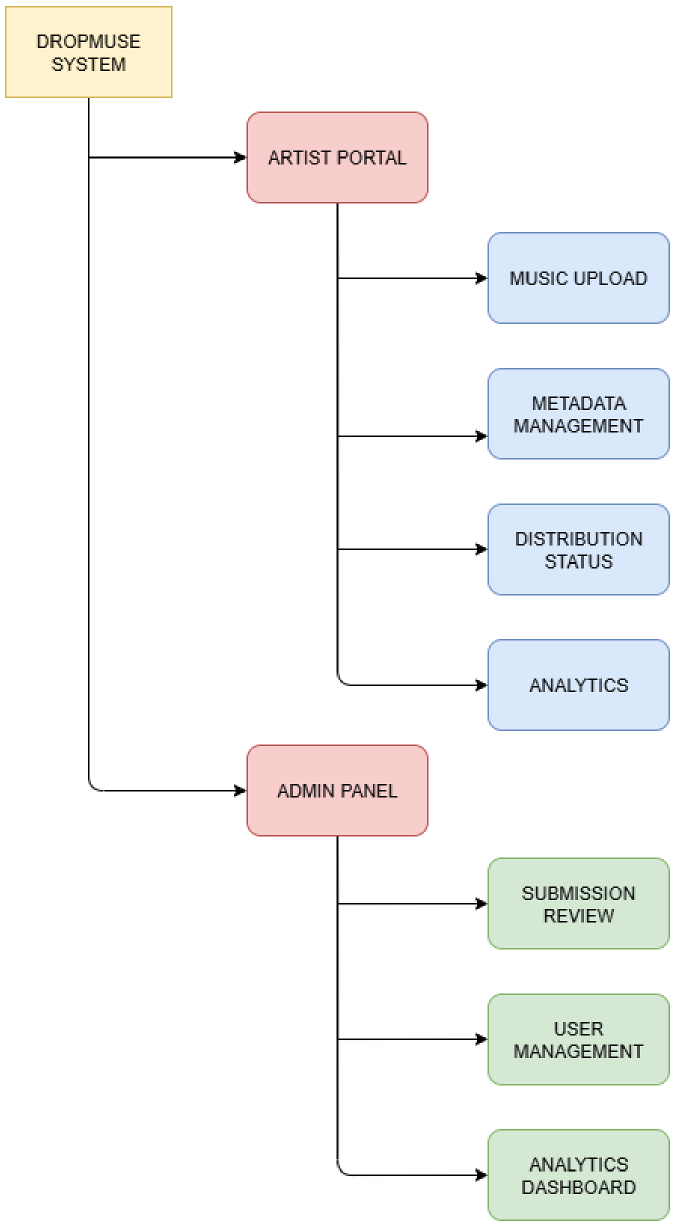

Figure 2: DropMuse System Overview

2.2 Product Functionality

- **User Registration and Authentication:** Allow artists and labels to create accounts and securely log in to the platform.
- **Music Upload:** Enable users to upload audio tracks in supported formats.
- **Metadata Management:** Allow artists and labels to manage the metadata of the songs
- **Distribution Management:** Allow artists and labels select stores and platforms where they want to distribute their music.
- **Dashboard and Analytics:** Allow artists and labels see their earnings and stats through friendly graphical interface.
- **Submission Review and Approval:** Allow administrators and moderators to review submitted content for compliance with platform and streaming service guidelines.
- **User Account Management:** Allow artists and labels to manage their account, change credentials and other information.
- **Multi Artist Management:** Allow labels to manage multiple artist accounts and distribute music on their behalf.
- **Support and Help Desk:** Provide users access to support channels for troubleshooting and assistance.

2.3 Design and Implementation Constraints

The development of **DropMuse** is subject to several constraints that influence both the design and implementation of the system. These constraints include:

- **Design Methodology:** The system must be designed following the COMET (Concurrent Object Modeling and Architectural Design) methodology, which guides the development of concurrent, distributed, and object-oriented systems.
- **Modeling Language:** UML (Unified Modeling Language) will be used to make different diagrams.
- **Technology Stack:**

  - **Frontend:** Must be implemented using HTML5, CSS3, JavaScript, and modern frameworks like React or Vue.js.
  - **Backend:** Should be built using Node.js or another JavaScript framework that supports RESTful APIs.
  - **Database:** Must use a scalable database system such as MongoDB or PostgreSQL.

- **Cloud Hosting:** The system must be hosted using Amazon Web Services (AWS) to ensure global scalability and security.
- **Security:** The system must comply with industry-standard security protocols such as HTTPS, OAuth 2.0 for API integrations, and encryption for sensitive data.
- **Streaming APIs:** Integration with third-party music platforms (e.g., Spotify, YouTube Music, Apple Music) should be done by considering their rate-limiting policies for APIs.
- **Cross-Platform Access:** The web application must be fully responsive and accessible across all modern desktop and mobile web browsers.

2.4 Assumptions and Dependencies

The following assumptions and dependencies are identified for the development of the **DropMuse** platform. If any of these assumptions prove to be invalid or the dependencies change, the design and implementation may be significantly affected:

- **Third-Party APIs:** It is assumed that all major streaming platforms (e.g., Spotify, Apple Music, YouTube Music) will maintain stable APIs for music distribution and metadata synchronization.
- **Internet Connectivity:** The platform assumes that users, moderators and administrators will have reliable internet access for uploading files, managing accounts, and communicating with external services.
- **Browser Compatibility:** The system assumes that users access the platform through modern web browsers that fully support HTML5, CSS3, and JavaScript (e.e., Chrome, Brave, Firefox).

- **Cloud Infrastructure Availability:** The application is dependent on AWS for storage, deployment, and scalability. It is assumed these services will remain available and affordable throughout development and operation.
- **User Compliance:** It is assumed that users will follow platform guidelines regarding content ownership and copyright compliance when uploading music.
- **Development Tools:** The team will rely on open-source libraries and tools for rapid development. It is assumed that these tools will remain compatible and well-documented.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The **DropMuse** platform offers a web-based graphical user interface (GUI) that is intuitive and user-friendly. Users interact with the system through a responsive dashboard using standard input devices such as mouse and keyboard, or touch on mobile devices.

The interface is designed with a modular layout, including the following key sections:

- **Login/Register Page:** Allows users to create accounts or sign in securely.
- **Dashboard:** Central panel where users upload tracks, manage their artist profile, view submission status, and access analytics.
- **Upload Interface:** Provides forms for metadata input (e.g., song title, genre, ISRC) and file upload.
- **Admin Panel:** Available to moderators/admins to review submissions, verify compliance, and manage users.

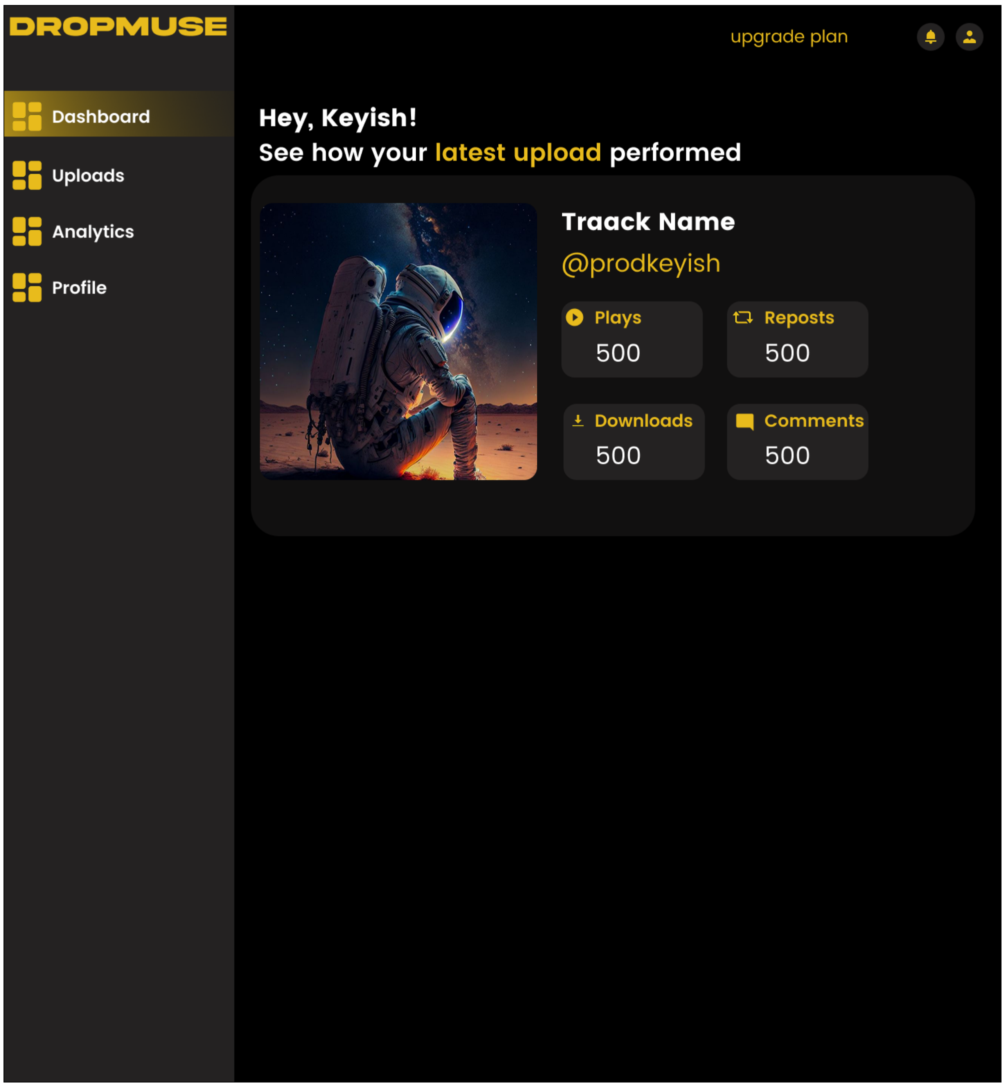

Figure 3: Mockup of DropMuse User Dashboard

3.1.2 Hardware Interfaces

The **DropMuse** platform does not directly interact with specialized hardware devices or sensors. However, it is designed to be compatible with commonly used consumer hardware for access and interaction.

Basic Hardware needs are:

- **Desktop and Laptop Computers:** Devices with a modern web browser (Google Chrome, Mozilla Firefox, Safari, Edge) and internet access. Input through keyboard and mouse.
- **Smartphones and Tablets:** Devices with touch interfaces running mobile web browsers (Android and iOS platforms).

- **Audio Interface:** Users may use local audio playback hardware to preview uploaded tracks; however, this does not affect platform functionality.

3.1.3 Software Interface

The DropMuse platform interfaces with several external software systems and services to perform its core functions. These interfaces include:

- **Streaming Service APIs:** DropMuse integrates with third-party music streaming platforms such as Spotify, Apple Music, and YouTube Music via their public APIs. These APIs are used to automate music publishing, update metadata, and retrieve distribution status.
- **Authentication Services:** The platform may utilize OAuth 2.0 or similar protocols to authenticate with external platforms and allow users to link their artist accounts on supported streaming services.
- **Internal Admin Tools:** The web application includes admin and moderator interfaces that connect with the backend database and content review modules to validate submissions before distribution.
- **(Optional) Mobile Companion App:** In future iterations, a mobile app may be developed to allow users to upload tracks, monitor submissions, and view analytics. This app would interact with the same backend API layer exposed by the main DropMuse platform.

3.2 Functional Requirements

R.1: User Registration and Login
Description: Enables artists and label managers to register, authenticate, and access the platform securely.

- **R.1.1: Register account**
  - Input: Username, email, password
  - Output: Account created and confirmation email sent
  - Processing: Store user details securely with password hashing
- **R.1.2: Login**

  - Input: Email, password
  - Output: User redirected to dashboard on success, error on failure
  - Processing: Validate credentials against stored records

R.2: Music Upload and Metadata Entry
Description: Allows users to upload audio files and enter necessary metadata.

- **R.2.1: Upload track**
  - Input: Audio file (MP3, WAV), cover art
  - Output: Track stored in user library
  - Processing: File validated, metadata extracted (optional), and saved
- **R.2.2: Enter metadata**
  - Input: Title, genre, artist name, album name, release date, language
  - Output: Metadata linked with uploaded track

R.3: Distribution to Streaming Services
Description: Enables submission of music to multiple platforms via API or admin moderation.

- **R.3.1: Select streaming platforms**
  - Input: List of desired platforms selected by user
  - Output: Selection stored for distribution
- **R.3.2: Submit for moderation**
  - Input: Uploaded track and metadata
  - Output: Sent for admin review
- **R.3.3: Distribute to platforms**
  - Input: Approved track and metadata
  - Output: Distributed via APIs to platforms like Spotify, Apple Music
  - Processing: API request formatted and sent

R.4: Admin and Moderator Panel
Description: Allows admins to review content and approve/disapprove submissions.

- **R.4.1: View submitted content**
  - Input: Database of uploaded content
  - Output: Moderation dashboard listing pending submissions
- **R.4.2: Approve/reject submissions**
  - Input: Moderation decision
  - Output: Track status updated, user notified

R.5: Analytics Dashboard
Description: Displays plays, reach, and revenue (if applicable) for each track.

- **R.5.1: View stats per platform**
  - Input: Request to fetch analytics
  - Output: Charts showing plays per service
- **R.5.2: View earnings**
  - Input: Request for revenue stats
  - Output: Tabular/graphical display of earnings

R.6: Label Management
Description: Labels can manage multiple artists from a single account.

- **R.6.1: Add artist profiles**
  - Input: Artist name, profile details
  - Output: Artist linked to label account
- **R.6.2: Upload and manage tracks per artist**
  - Input: Track uploads and metadata
  - Output: Stored under artist identity

R.7: Profile and Account Management

Description: Allows users to manage personal details and platform integrations.

- **R.7.1: Edit profile**
  - Input: New name, profile image, password
  - Output: Updated account settings
- **R.7.2: Manage streaming service links**
  - Input: API key/token or OAuth connection
  - Output: Linked accounts saved for automated publishing

3.3 Use Case Model

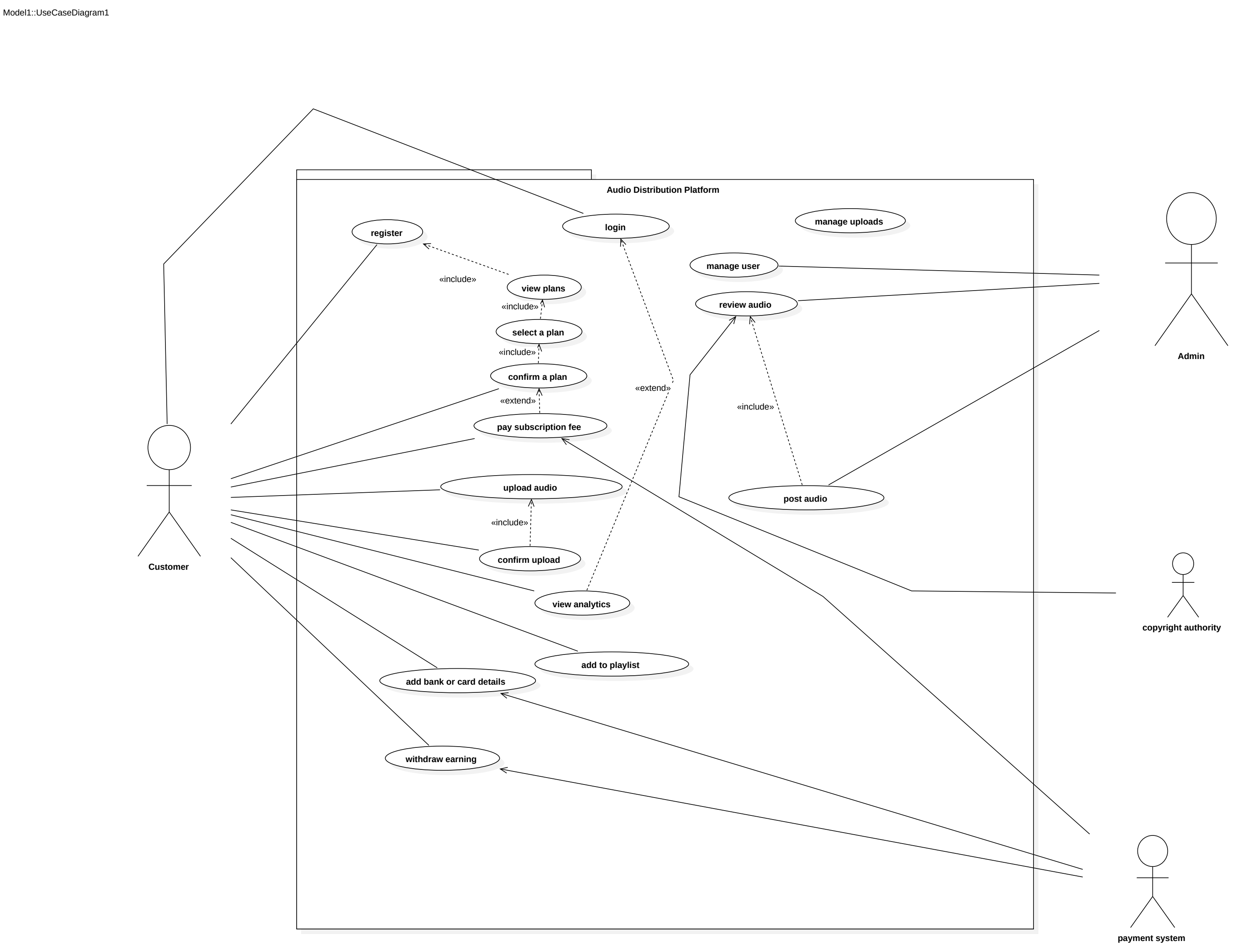

Figure 4: Use Case Diagram

3.3.1 Use Case U1: Register

Author: Aviskar
Purpose: To register a user
Priority: High
Post conditions: User's account should be created
Actors: Customer **Includes:** View Plan, Select Plan, Confirm Plan

Extends: Pay Subscription fee
Flow of Events:

- Click Register
- View Plan
- Select Plan
- Confirm Plan
- Pay fees if the plan is premium

3.3.2 Use Case U2: Upload Audio

Author: Aviskar
Purpose: To upload a track by user
Priority: High
Preconditions: User's account should be created and use should be on a plan
Actors: Customer **Includes:** Confirm Upload

3.3.3 Use Case U3: Manage User

Author: Aviskar
Purpose: To manage user and control info
Priority: High
Preconditions: Should have admin/moderator privilege
Actors: Admin

3.3.4 Use Case U4: Review Audio

Author: Aviskar
Purpose: To review users submitted track
Priority: High
Preconditions: Should have admin/moderator privilege
Actors: Admin **Include:** Post audio

3.4 Other System Model

3.4.1 Data Flow Diagram

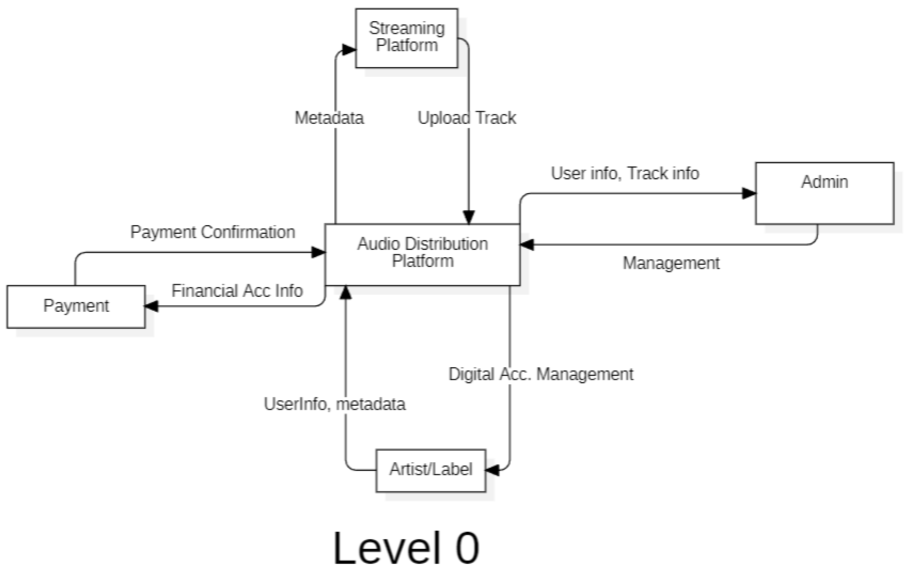

Figure 5: Level 0 DFD

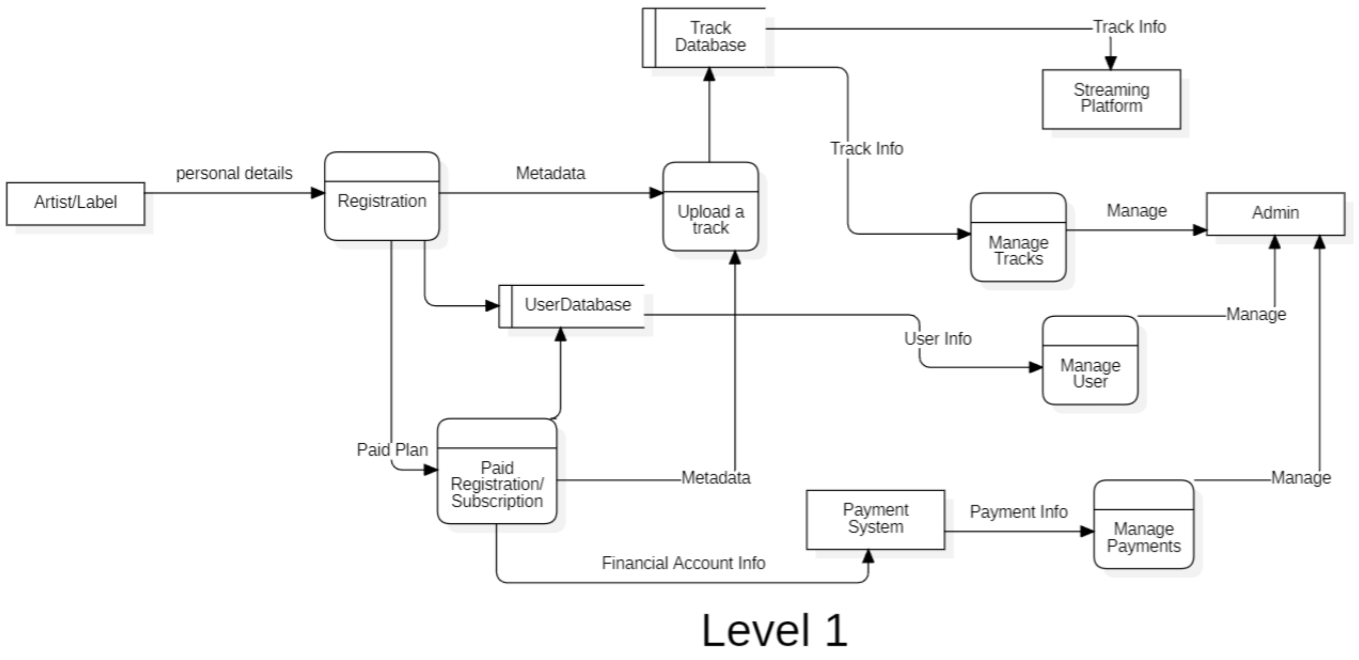

Figure 6: Level 1 DFD

4 Other Non-functional Requirements

4.1 Performance Requirements

- **P1.** The system shall respond to user requests for Analytics within 5 seconds under normal network conditions to ensure a smooth user experience.
- **P2.** The search functionality shall return results for queries of his uploads within 2 seconds.
- **P3.** The music upload process shall complete within 30 seconds for files up to 100 MB, depending on network speed.

4.2 Safety and Security Requirements

- **S1.** All user data including personal information and payment details shall be encrypted both in transit and at rest to prevent unauthorized access.
- **S2.** The system shall implement two-factor authentication for user accounts to strengthen identity verification.
- **S3.** The system shall enforce role-based access control (RBAC) to restrict administrative functions only to authorized personnel.

4.3 Software Quality Attributes

4.3.1 Reliability

- The system shall maintain 99.9% uptime annually, with planned maintenance windows announced at least 48 hours in advance.
- Regular backups of critical data will be performed daily to prevent data loss.

4.3.2 Adaptability

- The system design shall allow integration with different music encoding formats (e.g., MP3, WAV, FLAC) without requiring major architectural changes.

- The architecture shall support plugging in various payment gateways and user authentication providers to accommodate different market needs.

4.3.3 Usability

- The system shall provide an friendly user interface that requires no more than 10 minutes of training for new users to perform core functions.
- User feedback mechanisms shall be incorporated to allow easy reporting of issues or suggestions.

4.3.4 Maintainability

- The codebase shall follow standardized coding conventions and include comprehensive documentation to facilitate easier maintenance.

5 Other Requirements

5.1 Scalability

The system shall support scaling in the future to accommodate increased user load and storage requirements.

5.2 Backup and Disaster Recovery

Daily backups of critical data shall be performed and securely stored in geographically distributed locations.

5.3 Legal and Regulatory Compliance

The system shall comply with all applicable legal and regulatory requirements, including international data protection laws such as the General Data Protection Regulation (GDPR), as well as relevant Nepali laws such as the Electronic Transactions Act, 2063

(2006) and the Privacy Act, 2075 (2018). All content uploaded to the platform must adhere to copyright and digital rights management policies.

Appendix A – Data Dictionary

Name	Type	Description	Use
username	String	Name of the user	Used in login and display
email	String	Unique email for account	Login and password recovery
password	String	Encrypted user password	Used in authentication
track_file	File (MP3)	Uploaded audio file	Input for music suggestions
feedback	String	Optional user input	Collected for UI/UX improvement

Table 2: Appendix A – Data Dictionary