

Software Requirement Specification


Project name: Open-License Media Web Application

Date: Feb 24, 2025

Version: 1.0.0

By: Chức Vũ Văn

Revision history			
Version	Author	Version description	Date completed
1.0.0	Chức Vũ Văn		Feb 24, 2025

Review history			
Approving party	Version approved	Signature	Date
Chức Vũ Văn	1.0.0		Feb 24, 2025


Approval history			
Reviewer	Version reviewed	Signature	Date
Chức Vũ Văn	1.0.0		Feb 24, 2025

Table of contents

1. Introduction.....	2
1.1. Product scope.....	2
1.2. Product value.....	2
1.3. Intended audience.....	2
1.4. Intended use.....	2
1.5. General description.....	3
2. Functional requirements.....	3
2.1 User Account Management.....	3
2.2 Media Search and Retrieval.....	3
2.3 System Workflows & Transactions.....	4
2.4 Administrative & Compliance Requirements.....	4
3. External interface requirements.....	4
3.1 User Interface Requirements.....	5
3.2 Hardware Interface Requirements.....	5
3.3 Software Interface Requirements.....	5
3.4 Communication Interface Requirements.....	5
4. Non-functional requirements.....	5
4.1 Security.....	5
4.2 Capacity.....	5
4.3 Compatibility.....	6
4.4 Reliability.....	6
4.5 Scalability.....	6
4.6 Maintainability.....	6
4.7 Usability.....	6
4.8 Other Non-Functional Requirements.....	6
5. Definitions and acronyms.....	6

1. Introduction

1.1. Product scope

The Open-License Media Web Application aims to provide a central platform for discovering and managing openly licensed media. By integrating with the Openverse API, the system allows users to search, browse, and retrieve media efficiently while ensuring proper license attribution. The key goals include:

- **Accessibility:** Making openly licensed media easily discoverable and reusable.
- **User Experience:** Providing an intuitive, feature-rich interface with advanced search and filtering.
- **Security & Scalability:** Ensuring secure user authentication and modular architecture for future growth.
- **Best Practices:** Following modern software engineering principles for maintainability and efficiency.

This product is designed to serve content creators, educators, and researchers who rely on open-license media for their work.

1.2. Product value

The application simplifies media discovery by offering a seamless search experience, advanced filtering, and license transparency. It reduces the time users spend searching for usable media while ensuring compliance with licensing terms. The scalable architecture and well-documented API also support future enhancements and integrations.

1.3. Intended audience

The platform is designed for:

- **Content Creators** (e.g., graphic designers, video editors) who need open-license media for projects.
- **Educators & Students** who require media for presentations and research.
- **Researchers & Developers** who need openly licensed content for data analysis, AI training, or software projects.

1.4. Intended use

Users can:

- Search and filter openly licensed media from the Openverse API.

- View detailed licensing information to ensure proper attribution.
- Manage recent searches for easy reference.
- Browse and preview media within the application.

Use Case Example: A content creator looking for Creative Commons images can quickly search, filter by license type, and download media for their project without legal concerns.

1.5. General description

The Open-License Media Web Application is a web-based platform designed for intuitive media discovery. Built with a scalable and modular architecture, it ensures seamless API integration, user authentication, and secure data handling while maintaining high performance and reliability.

2. Functional requirements

2.1 User Account Management

- **User Registration & Authentication**
 - Users must be able to register an account using email and password.
 - The system must enforce password complexity rules.
 - If a user forgets their password, they must be able to reset it via email.
 - Users must be able to log in and log out securely.
 - If a user enters incorrect credentials **three times**, the account should be temporarily locked for security.
- **User Data Management**
 - The system must securely store and encrypt user data.
 - Users must be able to update their profile information.
 - Users should have the ability to delete their accounts permanently.
- **Recent Search Management**
 - Users must be able to save, retrieve, and delete recent searches.
 - The system should automatically delete search history after **30 days** unless saved by the user.

2.2 Media Search and Retrieval

- **Search Functionality**
 - Users must be able to search for media using keywords.
 - The system must integrate with the **Openverse API** to fetch search results.
 - If a search yields no results, the system must display a message with suggestions.
- **Filtering & Sorting**
 - Users must be able to filter search results by:

- **License type** (e.g., CC BY, CC0, Public Domain)
 - **Media type** (image, video, audio)
 - **Source** (specific repositories, e.g., Flickr, Wikimedia)
- Users should be able to sort results by relevance, date, or popularity.
- **Media Preview & Details**
 - Users must be able to preview media before downloading.
 - Each media item must display license details and attribution requirements.
 - Users should be able to view additional metadata (e.g., resolution, format, source).

2.3 System Workflows & Transactions

- **Search Workflow**
 - When a user submits a search query:
 1. The system sends a request to the Openverse API.
 2. The API returns media results.
 3. The system processes and displays the results with relevant metadata.
- **Download Workflow**
 - When a user clicks **Download**, the system must:
 1. Verify the availability of the media.
 2. Display a confirmation modal with licensing details.
 3. Redirect the user to the original media source for download.

2.4 Administrative & Compliance Requirements

- **Security & Compliance**
 - The system must use **OAuth 2.0** for secure authentication.
 - All user data must be encrypted and stored securely.
 - API keys and credentials must be stored securely and not exposed in frontend code.
- **Performance Requirements**
 - The system must return search results within **3 seconds** under normal load.
 - The application must support at least **100 concurrent users** without degradation in performance.
 - The system should cache frequent search queries to improve response times.
- **Logging & Error Handling**
 - All API requests and responses should be logged for debugging.
 - If an API request fails, the system should display an appropriate error message (e.g., "Service unavailable").
 - Users should be notified of failed actions (e.g., "Failed to save search history").

3. External interface requirements

3.1 User Interface Requirements

- **Usability:** The UI must be intuitive, with essential features accessible within **two clicks**. Onboarding guides and tooltips should be available for first-time users.
- **Responsiveness:** The UI should adapt to devices with screen sizes above **1024x768** and function well on desktops, tablets, and smartphones.
- **Performance:** The homepage must load in **3 seconds**. UI interactions (e.g., search, preview) should respond within **1 second**.

3.2 Hardware Interface Requirements

- **Supported Devices:** The system must be compatible with **Windows, macOS, Linux, iOS, and Android** devices with at least **2GB RAM** and **1GHz CPU**.
- **Network Connectivity:** The application should work under standard network conditions (Wi-Fi, 4G) and perform reasonably well under **low bandwidth** (e.g., 2G, 3G).

3.3 Software Interface Requirements

- **Openverse API Integration:** The system must use **HTTPS** to interact with the **Openverse API**, supporting structured responses (e.g., JSON) and handling **rate limits**.
- **Database Integration:** The system must use a relational database (e.g., **PostgreSQL, MySQL**) with **ACID compliance** for data integrity and regular backups.

3.4 Communication Interface Requirements

- **Email Integration:** The system must send basic confirmation emails (e.g., registration, password reset).
- **Error Notifications:** Display clear error messages for users and log critical errors.
- **Support:** Provide a simple contact form for user inquiries.

4. Non-functional requirements

4.1 Security

- All data transfers must use **HTTPS**.
- User passwords must be stored with **encryption**.
- User authentication must use **OAuth 2.0**.

4.2 Capacity

- The system should support **500 active users** concurrently.

- Data must be stored in a scalable database.

4.3 Compatibility

- The system must support **Windows, macOS, Linux, iOS, and Android**.
- It should work on the latest versions of **Chrome, Firefox, Safari, and Edge**.

4.4 Reliability

- The system must have **99.9% uptime**.
- Critical failures must be restored within **1 hour**.

4.5 Scalability

- The system should handle **1000 concurrent users** and be able to scale vertically and horizontally.

4.6 Maintainability

- The system should use **modular code** with **continuous integration** for easy updates and bug fixes.

4.7 Usability

- The system should have an intuitive, **responsive interface** for both desktop and mobile devices.

4.8 Other Non-Functional Requirements

- **Performance:** Search queries and media previews should load within **3 seconds**.
- **Regulatory:** The system must comply with **GDPR** and data privacy laws.

5. Definitions and acronyms