# **Software Requirements Specification (SRS) for AssetHub**

### **Table of Contents**

1. Introduction
2. Overall Description
3. Functional Requirements
4. Non-Functional Requirements
5. System Models

## **1. Introduction**

### **1.1 Purpose**

The purpose of this document is to define the requirements for **AssetHub**, a digital asset management system for MediaForge, which specializes in managing multimedia production, licensing, marketing, and archiving. The system will support the ingestion, management, versioning, and licensing of various multimedia assets (audio, video, image, and interactive files) and facilitate the unique workflows and access control needs of different departments within the company.

### **1.2 Scope**

**AssetHub** will be a web-based application with real-time and asynchronous processing capabilities. It will provide:

* Asset management for different media types (audio, video, image, interactive)
* A flexible licensing management system
* Department-specific workflows
* Version control with dependency tracking
* Role-based access control

### **1.3 Definitions, Acronyms, and Abbreviations**

* **DAM**: Digital Asset Management
* **SRS**: Software Requirements Specification
* **Asset**: A digital file that contains media (e.g., audio, video, image)
* **Dependency**: A relationship where one asset requires another to function correctly
* **Role-Based Access Control (RBAC)**: Access management based on the user's role

### **1.4 References**

* **IEEE 830-1998 SRS Standard**

## **2. Overall Description**

### **2.1 Product Perspective**

**AssetHub** will integrate with MediaForge’s existing storage and media servers. It is designed as a microservices architecture, allowing modular deployment of features such as asset management, licensing, workflow management, and access control.

### **2.2 Product Functions**

* **Asset Management**: Handle media files, metadata, and dependencies.
* **Licensing Management**: Assign licenses to assets with specific permissions and expiration.
* **Workflow Management**: Support workflows for production, marketing, and archiving.
* **Access Control**: Provide role-based access to features and assets.
* **Version Control**: Track versions of assets with dependency management.

### **2.3 User Characteristics**

* **Production Team**: Manages content creation and review.
* **Marketing Team**: Accesses selected assets for campaigns.
* **Archiving Team**: Manages long-term storage of assets.
* **Admin Users**: Full access to manage assets, users, and licenses.

### **2.4 Constraints**

* **Security**: Role-based access control is mandatory to avoid unauthorized access.
* **Scalability**: The system must handle thousands of assets and user requests concurrently.

## **3. Functional Requirements**

### **3.1 Asset Management**

#### **3.1.1 Asset Ingestion**

1. The system shall allow users to upload media assets (audio, video, image, interactive).
2. The system shall validate media format and metadata based on asset type.

#### **3.1.2 Metadata Management**

1. Each asset shall have metadata fields such as Title, Owner, Date Created, and Description.
2. Each asset type shall have specific metadata requirements:
   * Audio assets must include Bitrate.
   * Video assets must include Resolution.
   * Image assets must include DPI.

#### **3.1.3 Dependency Management**

1. The system shall allow users to define dependencies between assets.
2. The system shall restrict deletion of assets with active dependencies unless approved by an admin.

#### **3.1.4 Asset Processing**

1. The system shall support type-specific processing, such as resizing for images, transcoding for videos, and normalization for audio.
2. The system shall support both real-time and asynchronous processing as configured per asset.

### **3.2 Licensing Management**

#### **3.2.1 License Assignment**

1. The system shall allow users to assign one or more licenses to an asset.
2. Each license shall have the following attributes:
   * License Type
   * Expiration Date
   * Platform Restrictions
   * Usage Permissions

#### **3.2.2 License Validation**

1. The system shall validate a license’s expiration before granting access to licensed assets.
2. The system shall prevent expired licenses from being used unless renewed or extended.

#### **3.2.3 Flexible Licensing**

1. The system shall support new license types to be added without major code changes.
2. The system shall enable users to define custom permissions within each license.

### **3.3 Workflow Management**

#### **3.3.1 Workflow Definition**

1. The system shall define default workflows for each department: Production, Marketing, and Archiving.
2. Each workflow shall have unique stages (e.g., review and approval for Production, campaign planning for Marketing, archiving for Archiving).

#### **3.3.2 Workflow Customization**

1. The system shall allow admin users to modify workflows to fit evolving department needs.
2. The system shall allow department managers to initiate custom workflows based on asset requirements.

#### **3.3.3 Workflow Status Tracking**

1. The system shall track and display the status of each workflow (e.g., In Progress, Paused, Completed).
2. The system shall notify users involved in a workflow stage upon completion of the previous stage.

### **3.4 Access Control (RBAC)**

#### **3.4.1 Role Assignment**

1. The system shall define roles: Manager, Freelancer, and Admin, each with different permissions.
2. The system shall restrict access based on user role:
   * Manager: Full access to departmental assets and workflows.
   * Freelancer: Limited access to assigned assets only.
   * Admin: Full access to all assets, licenses, and workflows.

#### **3.4.2 Role-Based Permissions**

1. The system shall allow admins to assign roles to users.
2. The system shall allow managers to assign freelancers access to specific assets or workflows.

### **3.5 Version Control**

#### **3.5.1 Version Tracking**

1. The system shall maintain version history for each asset.
2. Each version shall include metadata like Version Number, Created Date, Changes, and Dependencies.

#### **3.5.2 Version Dependencies**

1. The system shall support version dependencies between assets, preventing updates that break dependent assets.
2. The system shall notify users when a new version affects dependent assets.