High Level Design Document

Introduction

This High Level Design (HLD) document outlines the architecture and core components of **MediaHub - Multi-Role Content Management Platform**. MediaHub is a sophisticated frontend application for media companies, enabling editors, writers, and admins to manage, schedule, and publish content efficiently. The platform is built with React, Redux Toolkit, and Tailwind CSS, supporting role-based dashboards, advanced media handling, and real-time collaboration.

---

1. System Architecture Overview

Architecture Description:
MediaHub is a modular, single-page application (SPA) structured around a component-based architecture. It leverages Redux Toolkit for state management and Tailwind CSS for UI styling. The system is designed for extensibility, maintainability, and secure role-based access.

Module/Component	Role/Responsibility
Authentication Module	Handles user login, registration, and role assignment
Dashboard Module	Provides role-specific dashboards (editor, writer, admin)
Content Management	Create, edit, schedule, and publish content
Media Handling	Upload, manage, and embed media assets
Collaboration Module	Real-time editing, comments, and notifications
State Management	Centralized state via Redux Toolkit
UI Layer	Responsive UI with Tailwind CSS and React components
API Integration Layer	Interfaces with backend APIs for data persistence

---

2. Component Interactions

Source Component	Target Component	Interaction Description
Authentication Module	Dashboard Module	Grants access based on user role
Dashboard Module	Content Management	Navigates to content creation/editing features
Content Management	Media Handling	Embeds or attaches media to content
Collaboration Module	Content Management	Enables real-time co-editing and comments
State Management	All Modules	Shares and synchronizes application state
API Integration Layer	All Modules	Fetches and persists data with backend

---

3. Data Flow Overview

Data Source	Data Destination	Data Type/Flow
User Input (UI Layer)	State Management	User actions, content edits, media uploads
State Management	API Integration Layer	API requests for CRUD operations
API Integration Layer	State Management	API responses, data updates
State Management	UI Layer	State-driven UI rendering
Collaboration Module	State Management	Real-time updates via WebSockets or polling

---

4. Technology Stack

Layer/Area	Technology/Frameworks
UI/Frontend	React, Tailwind CSS, HTML, CSS
State Management	Redux Toolkit, Redux
Language	TypeScript, JavaScript
Real-Time Collaboration	WebSockets (client-side)
API Integration	RESTful APIs (assumed)
Testing	Jest, React Testing Library (optional)

---

5. Scalability & Reliability

- **Scalability:**
  Modular architecture enables independent scaling of features. Efficient state management and lazy loading optimize performance for large user bases.
- **Reliability:**
  Robust error handling, optimistic UI updates, and state persistence ensure a stable user experience.
- **Security:**
  Role-based access control at both UI and API levels. Secure handling of authentication tokens and user data.

End of Document