Product Requirements & Specification Document

Project Name

MediaHub - Multi-Role Content Management Platform

Description

MediaHub is a robust frontend platform for media companies, enabling editors, writers, and admins to manage, schedule, and publish content. Built with React, Redux Toolkit, and Tailwind CSS, it features role-based dashboards, advanced media handling, and real-time collaboration.

---

1. Goals & Objectives

Goal	Description
Centralized Content Management	Unified interface for content creation, editing, and publishing.
Role-Based Access	Tailored dashboards and permissions for editors, writers, admins.
Advanced Media Handling	Upload, preview, and manage images, videos, and documents.
Real-Time Collaboration	Enable simultaneous editing and live updates.
Open-Source & Extensible	Modular, maintainable, and community-friendly architecture.

---

2. User Roles & Permissions

Role	Permissions
Admin	Full access: manage users, roles, settings, all content.
Editor	Approve, edit, schedule, and publish content; manage media.
Writer	Create, edit, and submit content drafts; limited media management.

---

3. Core Features

Feature	Description
Authentication	Secure login/logout, JWT-based session management.
Role-Based Dashboards	Custom UI per role, showing relevant actions and analytics.
Content Editor	Rich text editor with markdown, media embedding, autosave.
Media Library	Upload, preview, organize, and delete media assets.
Scheduling & Publishing	Set publish dates, status (draft, scheduled, published), and notifications.
Real-Time Collaboration	Live editing, presence indicators, and conflict resolution.

Activity Log	Track changes, user actions, and content history.
User Management	Admins can add, remove, and assign roles to users.
Responsive Design	Fully responsive UI for desktop and mobile.

---

4. Technical Specifications

Area	Specification
Framework	React (v18+), Redux Toolkit, TypeScript, Tailwind CSS
State Management	Redux Toolkit (slices for users, content, media, collaboration)
Styling	Tailwind CSS, custom utility classes
Routing	React Router v6+
API Integration	RESTful API (assumed backend), Axios for HTTP requests
Real-Time	WebSockets (e.g., Socket.IO) for collaboration features
Media Handling	File uploads via REST endpoints, client-side previews, drag-and-drop support
Testing	Jest, React Testing Library
Accessibility	WCAG 2.1 compliance, keyboard navigation, ARIA attributes
Internationalization	i18n-ready (English default)

---

5. UI/UX Requirements

- **Navigation:** Sidebar with role-based menu items.
- **Content Editor:** WYSIWYG + markdown, media embed, autosave, versioning.
- **Media Library:** Grid/list view, filters, bulk actions.
- **Collaboration:** User presence, live cursors, change indicators.
- **Notifications:** Toasts for actions, scheduled publish reminders.
- **Theming:** Light/dark mode toggle.

---

6. Non-Functional Requirements

Requirement	Specification
Performance	<2s load time, optimized asset delivery
Security	XSS/CSRF protection, secure file uploads
Scalability	Modular components, support for >1000 users
Maintainability	Clean code, documentation, open-source license
Extensibility	Plugin/module system for future features

---

7. Key Screens & Components

Screen/Component	Description
Login/Register	Auth forms, password reset
Dashboard	Role-based overview, analytics
Content List/Detail	Filter, search, view, and edit content
Content Editor	Rich editor, media embed, version history
Media Library	Upload, preview, organize media
User Management	List, add, edit, remove users (admin only)
Activity Log	View recent actions and changes
Settings	Profile, preferences, theme, notifications

---

8. Milestones & Deliverables

Milestone	Deliverable
M1: Project Setup	Repo, CI/CD, base scaffolding
M2: Auth & Role System	Auth flows, role-based routing
M3: Content Management	Editor, content list, scheduling
M4: Media Library	Upload, preview, management
M5: Collaboration	Real-time editing, presence
M6: User/Admin Features	User management, activity log
M7: Testing & QA	Unit/integration tests, accessibility checks
M8: Documentation	User/dev docs, open-source guidelines

---

9. Open Issues & Risks

- Real-time collaboration complexity
- Media upload scalability
- Role/permission edge cases
- API/backend dependencies

---

10. Appendix

Example Redux Slice (Pseudocode)

```
// contentSlice.ts
createSlice({
  name: 'content',
```

```
  initialState: { items: [], status: 'idle' },
  reducers: {
    addContent, updateContent, deleteContent, setStatus
  }
});
```

End of Document