Low Level Design Document

Introduction

This Low Level Design (LLD) document outlines the core implementation details for **MediaHub - Multi-Role Content Management Platform**. MediaHub is a React-based frontend enabling editors, writers, and admins to manage, schedule, and publish media content with role-based dashboards, advanced media handling, and real-time collaboration.

---

1. System Components

Component	Description / Responsibility
Auth Module	Handles user authentication and role-based access
Dashboard	Role-specific UI for editors, writers, admins
Content Manager	CRUD operations for articles/media, scheduling, publishing
Media Handler	Upload, preview, and manage media assets
Collaboration Layer	Real-time editing and notifications
State Management	Global state via Redux Toolkit
UI Layer	Responsive UI with Tailwind CSS

---

2. Class/Interface Overview

Class/Interface	Description / Key Methods/Attributes
`IUser`	User model: `id`, `name`, `role`, `token`
`IContentItem`	Content: `id`, `title`, `body`, `status`, `authorId`
`IMediaAsset`	Media: `id`, `url`, `type`, `uploadedBy`
`AuthService`	`login()`, `logout()`, `getCurrentUser()`
`ContentService`	`fetchContent()`, `create()`, `update()`, `delete()`
`MediaService`	`upload()`, `delete()`, `getMediaList()`
`CollabService`	`subscribe()`, `broadcastChange()`

Relationships:

- `Dashboard` uses `AuthService` for role checks
- `ContentManager` uses `ContentService`, `MediaService`
- `CollabService` interacts with content editing components

---

3. Data Structure Overview

Model	Fields
User	`id: string`, `name: string`, `role: 'admin'`
ContentItem	`id: string`, `title: string`, `body: string`, `status: 'draft'`
MediaAsset	`id: string`, `url: string`, `type: 'image'`

---

4. Algorithms / Logic

Role-Based Access Control (RBAC):

```
function canEditContent(user: IUser, content: IContentItem): boolean {
  if (user.role === 'admin') return true;
  if (user.role === 'editor') return true;
  if (user.role === 'writer' && content.authorId === user.id) return true;
  return false;
}
```

Content Publishing Flow:

```
On Publish Request:
  1. Validate user permission (RBAC)
  2. Validate content completeness
  3. Update content status to 'published'
  4. Notify collaborators via CollabService
```

---

5. Error Handling

Scenario	Handling Approach
Auth failure/expired token	Redirect to login, show error message
Unauthorized action (RBAC)	Show “Access Denied” notification
Media upload failure	Show error, allow retry
Network/API errors	Show generic error, retry option
Real-time sync/collab failure	Show warning, attempt reconnection

---

End of Document