Backend Developer Task Sheet – Video Editing Platform API

Project Title:

Build the Backend for a Web-based Video Editing Platform

Objective:

Create a scalable and modular backend service that allows users to upload videos, apply editing operations (trimming, subtitle overlay, audio modification, text/image addition), and download the rendered video. The goal is to handle video transformation using APIs, leveraging `FFmpeg` and persistent storage.

This task is focused solely on the **backend**. Frontend interaction is not required, but your API should be designed as if it will be consumed by a video editor frontend.

Tech Stack (Strict Requirements)

You **must** use:

- **Node.js**
- **Express.js**
- **PostgreSQL** (using Prisma ORM or Sequelize)
- **FFmpeg** (via `fluent-ffmpeg` or direct shell commands)
- **Multer** or equivalent for file uploads

Optional/Recommended:

- **Cloud storage**: AWS S3 or mock local storage

- **BullMQ / Redis** (for background rendering jobs)
- **Swagger / Postman** for API docs

Core Features to Implement

1. Video Upload Endpoint

- POST `/api/videos/upload`
- Accepts a video file (e.g., `.mp4`, `.mov`)
- Stores metadata in the DB (video name, duration, size, status)
- Save video file locally or in S3

Video Trimming / Cutting

- POST `/api/videos/:id/trim`
- Accepts start/end timestamps
- Uses FFmpeg to create a trimmed version
- Save trimmed video path and update DB

Add Subtitles

- POST `/api/videos/:id/subtitles`
- Accept subtitle text + start/end time
- Overlay on video using FFmpeg

6. Render Final Video

- POST `/api/videos/:id/render`
- Combines all changes into one final video
- Saves it and updates status in DB
- Optional: Trigger render via queue

7. Download Final Video

- GET `/api/videos/:id/download`
- Returns final rendered file for download

Submission Requirements

To complete your submission, please provide the following:

1. **GitHub Repository Link**
   Upload your project to a public or private GitHub repository. Make sure the repository:
   - Contains all necessary source code and assets
   - Includes a clear `README.md` with setup instructions and any relevant notes
   - Has clear commit history that reflects your progress
2. **Google Drive Link with Demo Video**
   Record a short screen recording (3–5 minutes) of your completed project demonstrating all key features. The video **must include a voice-over** explanation walking through:
   - The main interface and functionality
   - How each feature works (e.g., uploads, editing UI, previews)
   - Any challenges or creative decisions made
3. Upload the video to Google Drive and share the link with public or restricted access (as preferred). Make sure sharing permissions are correct.

Deadline

The final submission is due **by May 1st at 6:00 PM IST**.

> ⚠️ **Important:** Late submissions **will not be accepted** under any circumstances. Please ensure both the GitHub repo and the Google Drive demo link are shared **before the deadline.**

🙋 Tips

- Think about how a real editing tool would consume your API.
- Keep your FFmpeg logic modular — future enhancements should be easy.

- Clean, consistent error handling and logging is a big plus.
- If using background jobs, simulate processing time.