YouTube Clone - MERN Stack Documentation

# Github Link:-

<https://github.com/noubahar123/youtube-clone.git>

# Project Overview

This project is a YouTube Clone built using the MERN stack (MongoDB, Express, React, Node.js). It supports user authentication, video upload, playback, like/dislike reactions, comments with edit/delete, and suggested videos.

# Features

- User Authentication (JWT + Cookies)  
- Upload and manage videos (title, description, category, thumbnail)  
- Like/Dislike system with database integration  
- Comment system with edit and delete  
- Suggested videos sidebar  
- Responsive UI with Tailwind CSS  
- Secure API endpoints with role/user validation

# Project Structure

Backend:  
- Express server in /backend/src  
- Models: User, Video, Channel, Comment  
- Routes: /api/auth, /api/videos, /api/comments  
  
Frontend:  
- React app in /frontend/src  
- Components: VideoPlayer, Header, Sidebar, Auth pages  
- Context: AuthContext  
- API wrapper: axios instance

# Database Schema

Main collections:  
1. Users { username, email, passwordHash }  
2. Channels { channelName, owner }  
3. Videos { title, description, videoUrl, thumbnailUrl, channel, likes[], dislikes[], views, category }  
4. Comments { text, videoId, userId, timestamps }

# Execution Procedure

1. Clone repository  
2. Run `npm install` in both backend and frontend  
3. Configure `.env` for backend (Mongo URI, JWT\_SECRET, PORT)  
4. Run backend: `npm run dev` inside /backend  
5. Run frontend: `npm start` inside /frontend  
6. Access app at http://localhost:3000

# API Endpoints (Important)

- POST /api/auth/signup — Register user  
- POST /api/auth/login — Login user  
- GET /api/videos — List videos  
- POST /api/videos — Upload video  
- POST /api/videos/:id/like — Like/unlike a video  
- POST /api/videos/:id/dislike — Dislike/undislike a video  
- GET /api/comments/:videoId — Get comments  
- POST /api/comments/:videoId — Add comment  
- PATCH /api/comments/:id — Edit comment  
- DELETE /api/comments/:id — Delete comment

## 📂 Project Structure /backend ├── models/ (Mongoose schemas) ├── routes/ (API routes: videos, users, comments, auth) ├── middleware/ (authRequired) ├── server.js (entry point) /frontend ├── src/ ├── components/ ├── context/ ├── pages/ ├── api/

# Testing Guide

You can test with Postman, ThunderClient, or curl.  
  
Examples:  
1. Register user:  
 curl -X POST http://localhost:5000/api/auth/signup -d '{"username":"test","email":"a@b.com","password":"123"}' -H "Content-Type: application/json"  
  
2. Like video:  
 curl -X POST http://localhost:5000/api/videos/<videoId>/like -H "Authorization: Bearer <token>"  
  
3. Add comment:  
 curl -X POST http://localhost:5000/api/comments/<videoId> -d '{"text":"Nice!"}' -H "Authorization: Bearer <token>" -H "Content-Type: application/json"

# Test Cases

1. Create user, login, ensure token is valid ✅  
2. Upload video, verify it appears in list ✅  
3. Like a video twice → should toggle (like count unchanged) ✅  
4. Dislike video after liking → should move from likes[] to dislikes[]✅  
5. Add comment → should appear immediately✅  
6. Edit comment → updated text appears✅  
7. Delete comment → removed from list✅  
8. Suggested videos exclude current video✅

# Troubleshooting

- 500 Internal Error: Check MongoDB connection string ✅  
- CORS errors: Ensure frontend axios points to correct backend URL ✅  
- Auth errors: Ensure JWT\_SECRET in .env matches backend ✅  
- Duplicate mongoose import: Ensure single import per file ✅

# Conclusion

This MERN YouTube Clone demonstrates full-stack development skills: authentication, RESTful API design, state management, frontend-backend integration, and database handling. It provides a strong base for expanding into a complete video platform.