Full Stack Drive-like App - Project Structure & Boilerplate

# Overview

This project is a full-featured file management app like Google Drive, with user authentication,   
file uploads (multiple formats), folder creation, file preview, drag & drop UI, and an admin panel for user monitoring.  
The tech stack uses React (frontend), TailwindCSS (styling), Node.js (backend), Express.js (API), and MongoDB (database).

# Features

Frontend:  
- User Authentication (Register, Login, Logout, Forgot Password, Change Password)  
- Upload files (pdf, docx, xlsx, csv, zip, images)  
- Upload folders (using <input type="file" webkitdirectory />)  
- File Preview (PDF, Images)  
- Drag & Drop UI (react-dropzone)  
- Sort, Search, and Filter files/folders  
  
Backend:  
- Express routes for Auth, File handling, Admin  
- JWT-based authentication  
- Multer for file uploads  
- MongoDB models for users, files, folders, and logs  
- Admin monitoring: user activity, uploaded content  
  
Admin Panel:  
- View all users  
- See uploaded files/folders by user  
- Monitor activity logs  
- Modify user status

# Backend Folder Structure

backend/  
├── config/  
│ └── db.js  
├── controllers/  
│ ├── authController.js  
│ ├── fileController.js  
│ └── adminController.js  
├── middleware/  
│ ├── authMiddleware.js  
│ └── errorHandler.js  
├── models/  
│ ├── User.js  
│ ├── File.js  
│ ├── Folder.js  
│ └── ActivityLog.js  
├── routes/  
│ ├── authRoutes.js  
│ ├── fileRoutes.js  
│ └── adminRoutes.js  
├── uploads/  
├── .env  
├── server.js  
└── package.json

# Frontend Folder Structure

frontend/  
├── public/  
├── src/  
│ ├── assets/  
│ ├── components/  
│ │ ├── Auth/  
│ │ ├── Drive/  
│ │ └── UI/  
│ ├── context/  
│ ├── pages/  
│ ├── services/  
│ ├── App.jsx  
│ ├── index.js  
│ └── index.css  
├── tailwind.config.js  
├── postcss.config.js  
├── package.json  
└── .env

# Backend Setup

npm init -y  
npm install express mongoose dotenv bcryptjs jsonwebtoken multer cors  
npm install nodemon --save-dev

Server.js

|  |
| --- |
| const express = require('express');  const mongoose = require('mongoose');  const cors = require('cors');  require('dotenv').config();  const authRoutes = require('./routes/authRoutes');  const fileRoutes = require('./routes/fileRoutes');  const adminRoutes = require('./routes/adminRoutes');  const app = express();  app.use(cors());  app.use(express.json());  // Routes  app.use('/api/auth', authRoutes);  app.use('/api/files', fileRoutes);  app.use('/api/admin', adminRoutes);  const PORT = process.env.PORT || 5000;  mongoose.connect(process.env.MONGO\_URI, { useNewUrlParser: true, useUnifiedTopology: true })  .then(() => {  app.listen(PORT, () => console.log(`Server running on port ${PORT}`));  })  .catch(err => console.log(err)); |

env:  
PORT=5000

MONGO\_URI=mongodb://localhost:27017/mydrive

JWT\_SECRET=my\_super\_secret