Portfolio Project — Full Documentation (Frontend + Backend + CMS + 2-Week Plan)

🏗️ Architecture Overview

This project is built using **two separate repositories** + a **custom-coded CMS** (not using existing headless CMS platforms). Everything — API, admin panel, content models — will be built from scratch.

🔹 Repositories

1. **Frontend Repo** – React / Next.js + Tailwind CSS
2. **Backend Repo (CMS + API)** – Node.js + Express.js OR Python Flask/FastAPI
3. **CMS** – Custom-built admin panel + content API + database

---

📁 Repo Structure

1️⃣ Frontend Repo

Tech: React / Next.js + Tailwind CSS

Responsibilities:

- Fetch dynamic content from the custom CMS backend
- Render pages: Home, About, Projects, Skills, Experience, Blog, Contact
- SEO + animations + UI polish

Suggested Structure:

```
src/

components/

pages/ or app/

lib/api/
```

hooks/

layouts/

utils/

2️⃣ Backend Repo (Custom CMS + APIs)

Tech: Node.js + Express.js (recommended) OR Python Flask/FastAPI **DB:** PostgreSQL / MongoDB / Supabase

Responsibilities:

- Full CMS implementation from scratch
- Admin dashboard for managing content
- CRUD APIs for each content type
- Image/file uploads
- Authentication for admin panel
- Contact form handling
- Serve data to frontend via REST API

🛠️ Custom CMS (Built From Scratch)

You will create a minimal but complete CMS similar to Strapi/Sanity-like features.

📌 CMS Features to Build

Admin Panel UI (React/Vite or Next.js)

- Login page (JWT-based auth)
- Dashboard
- CRUD pages for:
    - About
    - Skills
    - Projects
    - Blogs
    - Experience/Timeline
    - Testimonials
    - Services

- Media upload system

Backend CMS APIs (Express or Flask)

Auth APIs

- POST /auth/login
- POST /auth/refresh

Content APIs

- /about (GET, PUT)
- /skills (GET, POST, PUT, DELETE)
- /projects (GET, POST, PUT, DELETE)
- /blogs (GET, POST, PUT, DELETE)
- /experience (CRUD)
- /testimonials (CRUD)
- /services (CRUD)

Media Upload APIs

- POST /upload/image → save to file storage

Contact Form APIs

- POST /contact → Save message + send email

📌 Database Schema (Custom)

Your CMS will store everything in your own database.

Example Tables

- users
- about
- skills
- projects
- blogs
- experience
- testimonials
- services
- messages (contact form)
- media

🔗 Data Flow

Frontend → Custom CMS API

Fetch content via REST API endpoints.

Admin Panel → CMS Backend

Full CRUD operations with authentication.

Contact Form Flow

Frontend → Backend → DB + Email

✨ Benefits of This Architecture

- 100% customizable CMS built from scratch
- Better control over security and logic
- Reusable for any client portfolio or business website
- No dependency on third-party CMS providers

🗓️ 2-Week Project Plan (Updated for CUSTOM CMS)

A complete revised plan to build CMS + Frontend portfolio.

📅 Week 1 – Backend CMS + Database + Admin Panel Setup

Day 1 — Backend + Database Setup

- Initialize Backend Repo
- Setup Express/Flask project
- Connect database (Postgres/Mongo)
- Create basic folder structure

Day 2 — CMS Authentication System

- Create Admin user model
- Implement login API (JWT)
- Setup middleware for protected routes

Day 3 — CMS Content Models (DB Schemas)

- Create models for (About, Skills, Projects, Blogs, Experience, Testimonials, Services)
- Build CRUD APIs for each

Day 4 — File Upload System

- Setup multer / cloud storage
- Create /upload endpoint

Day 5 — Admin Panel (Frontend)

- Setup Admin Panel using React or Next.js
- Build Login page
- Build Dashboard

Day 6 — CMS CRUD Screens

- Add pages for managing:
    - About
    - Skills
    - Projects

Day 7 — Remaining CMS Pages

- Add pages for:
    - Blogs
    - Testimonials
    - Experience
    - Services
- Fix UI

📅 Week 2 – Portfolio Frontend, Integration & Deployment

Day 8 — Portfolio Frontend Setup

- Initialize Frontend Repo
- Setup Tailwind
- Build layout + home page

Day 9 — Fetch Content From CMS

- Connect to backend APIs
- Display About, Skills, Projects dynamically

Day 10 — Additional Sections

- Blog page
- Testimonials
- Experience timeline

Day 11 — Contact Form

- Build UI
- Connect POST /contact backend API

Day 12 — Deployment (Backend + CMS Admin)

- Deploy backend to Render/Railway
- Setup environment variables
- Deploy Admin Panel

Day 13 — Deploy Portfolio Frontend

- Deploy to Vercel/Netlify
- Configure domain + API URLs

Day 14 — Final Testing & Optimization

- API validation
- Image optimizations
- Security checks
- Polish UI + fix bugs

📌 Final Deliverables

Frontend Repo:

- Portfolio UI fully connected to custom CMS
- Fully responsive + SEO optimize
- CMS Frontend

Backend Repo (CMS):

- Custom admin panel
- REST APIs for all content models
- DB + authentication + file uploads
- Contact form save + email

Database:

- All tables for content management