Portfolio Hub Pro - Backend Integration Guide

This document provides comprehensive instructions for the backend developer to integrate the **Portfolio Hub Pro** frontend with a production-ready API.

---

1. Project Overview

Portfolio Hub Pro is a modern, full-stack portfolio management system built with **React**, **TypeScript**, and **Tailwind CSS**. It features a public-facing portfolio and a secure admin dashboard for content management.

Tech Stack

- **Frontend:** React 18, Vite, TypeScript, Tailwind CSS.
- **State Management:** React Context API & TanStack Query.
- **Icons:** Lucide React & Ionicons.
- **Real-time:** Laravel Echo & Pusher (Optional).

---

2. Frontend Setup Instructions

Prerequisites

- **Node.js** (v18.0.0 or higher)
- **npm** or **yarn**

Installation

1. Clone the repository.

2. Navigate to the project directory:

```
cd portfolio-hub-pro
```

3. Install dependencies:

```
npm install
```

Running the Project

- **Development Mode:**

```
npm run dev
```

- **Production Build:**

```
npm run build
```

---

3. API Integration Details

Switching to Production API

By default, the project runs in **Mock Mode**. To connect to the real backend:

1. Open `src/api/request.js`.
2. Set `MOCK_MODE = false` (Line 3).
3. Update the `BASE_URL` in `src/api/endpoints.js` to your backend URL (Default: `http://localhost:8000/api`).

Authentication

The frontend uses **JWT (JSON Web Tokens)** for authentication.

- **Storage:** Tokens are stored in `localStorage` under the key `auth_token`.
- **Headers:** All protected requests include the `Authorization: Bearer <token>` header.

---

4. Required API Endpoints

The backend must implement the following RESTful endpoints:

Category	Method	Endpoint	Description
Auth	POST	`/auth/login`	Returns JWT token and user object
Settings	GET	`/settings`	Fetch site settings (Logo, Socials)
Settings	POST	`/settings/update`	Update site settings
Profile	GET	`/profile`	Get personal info (Bio, Job, CV)
Profile	POST	`/profile/update`	Update profile (Supports FormData)
Resume	GET	`/resume`	Get full resume (Skills, Exp, Edu)
Resume	POST	`/resume/update`	Update resume structure
Portfolio	GET	`/portfolio`	List all projects
Portfolio	POST	`/portfolio`	Create project (Supports FormData)
Portfolio	PUT	`/portfolio`	Update project details
Portfolio	DELETE	`/portfolio`	Delete project
Blog	GET	`/blog`	List all blog posts
Blog	POST	`/blog`	Create post (Supports FormData)
Blog	DELETE	`/blog`	Delete blog post
Testimonials	GET	`/testimonials`	List client feedback
Testimonials	POST	`/testimonials`	Add new testimonial
Testimonials	DELETE	`/testimonials`	Remove testimonial
Messages	GET	`/messages`	List contact form submissions
Contact	POST	`/contact`	Public endpoint to send messages

---

5. Data Structures (Expected JSON)

Testimonial Object

```
{
  "id": 1,
  "name": "Client Name",
  "avatar": "https://url-to-image.com",
  "text": "Feedback content here...",
  "date": "2024-01-08"
}
```

Note: The `rating` field has been removed as per the latest requirements.

Project Object

```
{
  "id": 1,
  "title": "Project Title",
  "category": "Web Development",
  "image": "https://url-to-image.com",
  "description": "Project details..."
}
```

---

6. Backend Developer Checklist

- [ ] Implement JWT Authentication.
- [ ] Enable **CORS** for the frontend domain.
- [ ] Handle **FormData** for file uploads (Avatars, Project Images, CV PDFs).
- [ ] Ensure all GET endpoints return data wrapped in a `data` property or as a direct array as handled in `request.js`.

Generated on: January 08, 2026