

Portfolio Dashboard Project Roadmap

Project Vision

Build a **Portfolio Dashboard Platform** where users can log in, fill in profile info, skills, and projects, choose a portfolio template, and publish their public portfolio page. Includes admin panel for management.

Learning Goals

- JWT Authentication + Refresh Tokens
 - CRUD Operations (Axios, manual error handling)
 - Redux Toolkit for state management
 - React Router (protected/public routes)
 - Hooks: useEffect, useMemo, useCallback, useRef
 - Context API for theme toggling
 - UI Layout and Styling (Navbar, Sidebar, Footer)
 - Error handling, caching, performance optimization
-

Step-by-Step Roadmap

Phase 1 — Setup & Structure

1. Initialize React App (Vite or CRA)
2. Setup folder structure:
3. `/components` (Navbar, Sidebar, Footer)
4. `/features` (Auth, UserProfile, Portfolios, Admin)
5. `/app` (Store, Context)
6. `/pages` (Login, Dashboard, Profile, Admin, PublicPortfolio)
7. Install dependencies: React Router, Redux Toolkit, Axios, Tailwind/MUI
8. Setup Routing:
9. `/` (Landing page)
10. `/login`
11. `/dashboard` (Protected)
12. `/admin` (Protected)
13. `/portfolio/:username` (Public)

Goal: Have basic routing and placeholder pages.

Phase 2 — UI & Layout

1. Design layout:
2. Navbar (logo, theme toggle, user avatar)
3. Sidebar (menu links)
4. Footer (simple info)
5. Responsive design: collapsible sidebar for mobile.
6. Theme toggle via Context API (persist with localStorage).

Goal: Static, responsive UI ready for backend integration.

Phase 3 — Authentication (JWT + Refresh Token)

1. Create Login/Register forms with validation.
2. Backend (FastAPI) issues:
3. Access Token (30 min expiry)
4. Refresh Token (7 days expiry)
5. Frontend logic:
6. Axios interceptors for attaching token
7. Auto-refresh token before expiry
8. Redux authSlice: manage `user`, `token`, `isAuthenticated`, etc.
9. Protect routes (redirect if no token).

Goal: Full auth flow with auto-refresh and protected routes.

Phase 4 — CRUD: Portfolio Data

User manages: - Profile Info - Skills (CRUD) - Projects (CRUD) - Template Selection

Pages: - Dashboard (overview) - Profile (forms) - Skills (CRUD) - Projects (CRUD) - Preview (portfolio preview)

Goal: CRUD operations with manual loading/error handling and Redux caching.

Phase 5 — Admin Panel

1. Admin Dashboard: view all users, delete/deactivate accounts.
2. Admin-only routes (role-based access).
3. UseMemo for computed summaries.

Goal: Role-based admin system.

Phase 6 — Public Portfolio Page

1. Route `/portfolio/:username`
2. Fetch and render public portfolio data.
3. Handle loading/errors gracefully.

Goal: Public, dynamic portfolio pages per user.

Phase 7 — UI Polish & Performance

1. Add loading spinners, toasts, and error boundaries.
2. Optimize with:
3. `useCallback` (for handlers)

4. useMemo (for filtered data)
5. useEffect (for fetching)
6. useRef (for DOM/timers)
7. Improve Redux caching and minimize re-fetching.

Goal: Smooth, responsive, optimized UI.

Phase 8 — Deployment & Testing

1. Deploy frontend (Vercel/Netlify) and backend (Render/Railway).
2. Configure CORS and environment variables.
3. Test login flow, token refresh, and CRUD ops.

Goal: Fully deployed, tested, and stable app.

Hook Usage Summary

Hook	Purpose
useEffect	Fetch data, handle side effects
useMemo	Cache derived/computed data
useCallback	Optimize event handlers
useRef	Manage focus, DOM refs, store previous state
Context API	Manage theme and global UI states
Redux Toolkit	Global state for auth, caching, UI

Final Stack

- **Frontend:** React (Vite) + Redux Toolkit + Context + Axios
 - **Backend:** FastAPI + JWT Auth + Refresh Tokens
 - **Features:** Auth, CRUD, Admin Panel, Public Portfolios, Theming, Caching, Protected Routes
-

Next Step: Start **Phase 1 — Setup & Structure** and build it step-by-step.