Portfolio App Summary

What it is

A personal portfolio web app built with Next.js 16, React 19, and TypeScript. It showcases design and development case studies from MDX content, with a protected resume viewer and download flow.

Who it is for

Primary audience: recruiters, hiring managers, and potential clients evaluating Hunter Bastian, who is described in repo metadata as a student product designer and photographer.

What it does

- Renders a home page of case studies sourced from local MDX files in content/projects.
- Generates per-project routes at /projects/[slug] with SEO metadata, hero media, tags, and optional demo/GitHub links.
- Provides an archive page that lists projects marked archived in frontmatter.
- Includes a resume modal that checks access, unlocks via password, and serves PDF inline or as download.
- Uses motion-driven UI patterns (section transitions, staggered reveals, responsive nav, and overlay interactions).
- Adds a footer Snake mini-game easter egg implemented entirely client-side.
- Registers a service worker for static asset caching, plus Vercel Analytics and Speed Insights hooks.

How it works (repo-evidence architecture)

- Content/data layer: src/lib/projects.ts reads content/projects/*.mdx via fs + gray-matter, then sorts/filter projects by frontmatter (including archived).
- App routing layer: src/app/page.tsx loads all projects for the home view; src/app/projects/[slug]/page.tsx loads one project and renders MDX with next-mdx-remote; src/app/archive/page.tsx renders archived projects.
- UI/component layer: AnimatedHomePage, ProjectGridClient, ProjectCard, Header/Footer, and ResumeModal compose the interactive front end.
- Resume service flow: /api/resume/status, /api/resume/unlock, and /api/resume/file use src/lib/resumeAuth.ts to issue and verify signed cookie tokens before returning private/resume/Hunter Bastian Resume.pdf.
- Platform/performance layer: Next.js config enables MDX, image optimization, standalone output, cache/security headers, and optional bundle analysis.
- Not found in repo: database, message queue/background workers, or third-party API integrations beyond Vercel analytics/speed insights.

How to run (minimal)

- Use Node.js >= 18.17.0.
- Install dependencies: npm install
- Optional: set RESUME_PASSWORD in .env.local to lock resume access (fallback default exists in code).
- Start dev server: npm run dev
- Open http://127.0.0.1:3000