Architecture Overview

Stack

Concern	Choice
Framework	**Next.js 16.2** (App Router), **React 19.2**, **TypeScript 5** (strict)
Styling	**Tailwind CSS v4**, with CSS-first `@theme` tokens in `src/app/globals.css` and no `tailwind.config.js`
Rendering	**Static export** (`output: "export"`, `trailingSlash: true`) → plain HTML/CSS/JS in `out/`
Hosting	**Cloudflare Pages**, Git-integrated; every push to `main` deploys
Server code	One **Cloudflare Pages Function**: `functions/api/event-inquiry.ts`
CMS	**Decap CMS 3.x** at `/admin`, GitHub backend, commits JSON to `content/`
Media	`public/` for committed images; **Cloudflare R2** (`cdn.r2.fortadams.org`) for video and PDFs
Tours	**FareHarbor** Auto-Lightframe (script plus booking URLs; no API)
CRM / payments	**HighLevel** (LeadConnector API, inbound webhooks, hosted payment links and forms)
Tooling	ESLint 9 (`eslint-config-next`), Prettier 3 with `prettier-plugin-tailwindcss`

> **Next.js 16 differs from older versions.** The repo's `AGENTS.md` warns that APIs, conventions, and file structure may differ from what you know. Read the bundled docs in `node_modules/next/dist/docs/` before using an unfamiliar API, and heed deprecation notices.

Request flow

```
Browser ──GET──▶ Cloudflare Pages CDN ──▶ out/**/index.html + /_next/static/*
   │
   ├─ POST /api/event-inquiry ──▶ Pages Function ──▶ HighLevel API
   │                                  (contacts/upsert, opportunities)
   ├─ POST (no-cors) ─────────────────▶ HighLevel inbound webhooks
   │                                  (volunteer, corporate, newsletter)
   ├─ fareharbor.com/embeds/api/v1/?autolightframe=yes  (booking overlay)
   ├─ cdn.r2.fortadams.org/video/*.mp4, /documents/*.pdf
   └─ links out to crm.fortadams.org (donation form, membership payment links)
```

Key design decisions

Static export, no Node server. Hosting is effectively free, fast, and has very little attack surface. Consequences:

- No Next.js image optimisation server. A custom loader (`src/lib/image-loader.ts`) passes URLs through, or routes them via Cloudflare Image Resizing when `NEXT_PUBLIC_CF_IMAGE_BASE` is set.
- No route handlers, server actions, ISR, or middleware. Dynamic behaviour is client-side, or lives in `functions/`.
- **Anything time-based is evaluated at build time.** For example, `getUpcomingEvents()` compares against `Date.now()` during the build, and the footer year comes from the build. Content only "ages" when the site rebuilds.
- `process.env.NEXT_PUBLIC_*` values are **inlined at build time** and are public. Changing them requires a rebuild.
- `sitemap.ts` and `robots.ts` use `export const dynamic = "force-static"`.

Content in Git. CMS-editable data is JSON in `content/`, imported statically by loaders in `src/lib/`. Everything else is typed TypeScript data in `src/lib/*.ts`. There's no database. Git history is the audit log and the backup.

Constrained CMS. Decap only edits a few JSON files with validated fields. Page layout and copy stay in code, so editors can't break the design.

Third parties own their domains. FareHarbor owns tours, pricing, and checkout. HighLevel owns contacts, payments, and subscriptions. The site holds URLs, not product data. Integrations are isolated in a few files so a provider can be swapped locally (see *HighLevel Integrations* and *FareHarbor Integration*).

One server-side secret path. The event inquiry needs a real success/failure response and a private API token, so it runs as a Pages Function. The other lead forms use fire-and-forget webhooks from the browser.

Revision #1
Created 2026-09-28 14:25:56 UTC by David Prata
Updated 2026-09-28 14:25:57 UTC by David Prata