> Notes

• The README references the /v1 contracts and cutover plan where IntegAI becomes the single API entry point for the site, including idempotent forms and artifact publishing.

• It includes the Globe, News & Trends, and Energy features you requested, plus practical Cursor/Gemini local‑dev tips from your notes.

• Visual directives (starfield, R3F, motion, MDX tone) are wired into structure and component guidance.

• CROx (Pharma) is called out with KPIs to anchor sector value.

---

# 🌌 HISL Website (Next.js 15.5 + IntegAI v1)

> \*\*Cinematic, sovereign, and fast.\*\* This repo powers the public HISL site using Next.js App Router, MDX content, R3F visuals, and an offline‑first backend orchestrated by \*\*IntegAI\*\*.

[![Node](https://img.shields.io/badge/node-%3E%3D20.x-339933?logo=node.js)]()

[![Next.js](https://img.shields.io/badge/Next.js-15.5-black?logo=next.js)]()

[![TypeScript](https://img.shields.io/badge/TypeScript-5.x-3178C6?logo=typescript)]()

[![Vercel](https://img.shields.io/badge/Deploy-Vercel-black?logo=vercel)]()

---

## TL;DR

- \*\*One surface:\*\* Website only calls \*\*IntegAI\*\* `/api/v1/site/\*` (read) and `/api/v1/site/forms/\*` (write).

- \*\*Performance guardrails:\*\* Cached reads target \*\*TTFB ≤ 200 ms\*\*; RAG reads \*\*≤ 2.0 s p95\*\*; LLM gen \*\*≤ 10 s p95\*\*.

- \*\*Sections, not pages:\*\* The site renders \*\*block‑structured page JSON\*\* `{sections[]: {sectionId, component, props}}`.

- \*\*Cinematic UI:\*\* R3F starfield/globe, Framer Motion, MDX content, graceful fallbacks.

- \*\*Show the math:\*\* Energy badge (Wh → kgCO₂e) on `/globe`; live trends on `/news`.

- \*\*Idempotent forms:\*\* POSTs require `Idempotency-Key`; returns `{ok, triage, route}`.

---

## Table of Contents

1. [Architecture Overview](#architecture-overview)

2. [Folder Structure](#folder-structure)

3. [Getting Started](#getting-started)

4. [Configuration (.env)](#configuration-env)

5. [NPM Scripts](#npm-scripts)

6. [API Contracts (v1)](#api-contracts-v1)

7. [Component Registry (sections → components)](#component-registry-sections--components)

8. [Pages & Features](#pages--features)

9. [SEO & Content](#seo--content)

10. [Performance Budgets](#performance-budgets)

11. [Telemetry & Security](#telemetry--security)

12. [CI/CD & Quality Gates](#cicd--quality-gates)

13. [Contributing](#contributing)

14. [License](#license)

15. [Appendix: Code Snippets](#appendix-code-snippets)

---

## Architecture Overview

```mermaid

flowchart LR

subgraph Web[HISL Website (Next.js 15.5)]

U[User] -->|GET /| APP[App Router]

APP -->|fetch| API[/api/v1/site/\*/]

end

API --> NGINX[NGINX Reverse Proxy]

NGINX --> GW[IntegAI Gateway]

subgraph IntegAI[IntegAI (offline‑first)]

GW --> L1[(L1 Cache)]

GW --> Planner[Planner]

Planner --> Retriever[Retriever: Chroma + Neo4j]

Retriever --> Judge[Judge]

Judge --> Executor[Executor: llama.cpp Gemma → Mistral]

Executor --> Safety[Safety]

Safety --> Publisher[Publisher → .integpkg (MinIO)]

Publisher --> L1

end

APP --> Forms[/api/v1/site/forms/{formId}/]

Forms --> NGINX --> GW

Cutover: proxy → cache → takeover; website never calls third‑party services directly—only IntegAI. (Details in contracts & execution plan below.)

---

Folder Structure

/app

/(marketing) # Home, About, Sectors, Agents, Poem

/globe # R3F globe demo + energy badge

/news # Live Trends (Grok) page

/api # (optional) local helpers; primary APIs are IntegAI

/components # UI components (RSC + Client)

/content # MDX content (bios, poem, longform)

/lib # api client, registry, utils

/public/imagery # starfields, earth, nebulae (processed via Sharp)

/scripts # ci, imagery, cache warmer, seo checks

/types # shared types (PagePayload, Section, SEO, Forms)

/e2e, /\_\_tests\_\_ # Playwright / Vitest

---

Getting Started

Prerequisites

Node 20+, pnpm 9+ (or npm/yarn)

Vercel (recommended) or your Node host

API: IntegAI gateway URL (staging/prod)

> Tip: For local DX, Cursor (v0.46+) with Composer mode worked well on Linux Mint; pair with a high‑context model for repo‑wide edits. (Optional.)

Install & Run

pnpm install

cp .env.example .env.local

pnpm dev # http://localhost:3000

pnpm build && pnpm start

Quick Smoke

Open / and /globe.

Submit /contact with an Idempotency-Key header; confirm success message.

---

Configuration (.env)

Create .env.local:

# Core

NEXT\_PUBLIC\_SITE\_URL=http://localhost:3000

# IntegAI endpoints (all website calls go here)

INTEGAI\_API\_BASE=https://site.hisl.ai/api

INTEGAI\_FORMS\_TIMEOUT\_MS=10000

# Telemetry

SENTRY\_DSN=

NEXT\_PUBLIC\_POSTHOG\_KEY=

NEXT\_PUBLIC\_POSTHOG\_HOST=https://eu.i.posthog.com

# Live features (optional for local)

GROK\_API\_KEY= # News & Trends

DEEPSEEK\_API\_KEY= # Globe demo (if using DeepSeek for copy)

# Energy badge defaults

NEXT\_PUBLIC\_ENERGY\_BASE\_WH=0.3

NEXT\_PUBLIC\_ENERGY\_PER100TOK\_WH=0.1

NEXT\_PUBLIC\_CO2\_KG\_PER\_KWH=0.233

> Secrets are never hardcoded; set them in Vercel env vars for deploys.

---

NPM Scripts

{

"scripts": {

"dev": "next dev",

"build": "next build",

"start": "next start -p 3000",

"typecheck": "tsc --noEmit",

"lint": "eslint .",

"fmt": "prettier -w .",

"test": "vitest run",

"test:e2e": "playwright test",

"imagery:manifest": "node scripts/generate-imagery-manifest.mjs",

"seo:check": "node scripts/seo-check.mjs",

"cache:prewarm": "node scripts/cache-prewarm.mjs",

"lhci": "lhci autorun"

}

}

---

API Contracts (v1)

All content is block‑structured. Website fetches pages via GET:

GET /v1/site/page?slug={slug} → { title, sections[], metadata, lastUpdated }

GET /v1/site/menu → {primary[], footer[]}

GET /v1/site/seo?slug={slug} → { title, description, keywords[], og, schemaOrg }

GET /v1/site/search?q=...&limit=&after= → { results[], next }

POST /v1/site/forms/{formId} (with Idempotency-Key) → { ok, idempotencyKey, triage, route }

Timeouts: 300 ms cached, ≤2.5 s non‑cached, ≤10 s gen. All POSTs are idempotent. Versioning: /v1 stable; break only under /v2.

---

Component Registry (sections → components)

Render strictly by sectionId → component → props mapping. Unknown props are rejected at build time.

// /lib/section-registry.ts

export type SectionId =

| 'nav\_primary\_v1' | 'hero\_cosmic\_v1' | 'perf\_badges\_v1'

| 'api\_contracts\_v1' | 'agents\_matrix\_v1' | 'feature\_tile\_v1'

| 'split\_feature\_v1' | 'big\_cta\_v1' | 'footer\_v1'

| 'hero\_globe\_v1' | 'prompt\_console\_v1' | 'arcs\_visual\_v1'

| 'energy\_badge\_v1' | 'result\_panel\_v1' | 'hero\_news\_v1'

| 'stream\_filters\_v1' | 'live\_stream\_v1' | 'annotation\_bar\_v1'

| 'bio\_longform\_v1' | 'poem\_body\_v1' | 'agents\_grid\_v1'

| 'kpi\_grid\_v1' | 'proof\_blocks\_v1';

export type Section<T extends SectionId = SectionId> = {

id: T;

component: string;

props: Record<string, unknown>;

order?: number;

};

export const registry: Record<SectionId, () => Promise<any>> = {

hero\_cosmic\_v1: () => import('@/components/sections/HeroCosmic'),

energy\_badge\_v1: () => import('@/components/sections/EnergyBadge'),

// ...etc

};

---

Pages & Features

Home /

Hero starfield, performance badges, API contracts strip, Agents matrix, Globe & News teasers, sovereign ethics block, big CTA.

Globe /globe

R3F globe, prompt console, prompt arcs visualization, energy badge (Wh → kgCO₂e), result panel with provenance.

Energy math:

Wh = baseWh + (tokens/100) \* per100TokWh (defaults: 0.3 Wh base, 0.1 Wh/100 tokens).

kgCO2e = (Wh / 1000) \* co2KgPerKWh (default 0.233 kg/kWh).

Respects prefers-reduced-motion.

News & Trends /news

Live stream powered by Grok (server action + streaming UI).

Filters (topics/sources), annotation bar (Pin/Save/Send to Agents).

About /about/michael, /about/integai and Poem /poem

MDX‑backed longform with styled typography and motion entrances.

Agents /agents + Sectors /sectors/\*

Directory of agents; CROx deep‑dive landing with feature columns, KPI grid, and auditable proof blocks.

Forms /contact, /rfp, /newsletter, /careers

POST via /v1/site/forms/{formId}; requires Idempotency-Key. Shows confirmation pages.

---

SEO & Content

API‑driven SEO via /v1/site/seo?slug; enforces one source of truth for titles/OG/schema.

MDX for bios, poem, essays: frontmatter controls SEO & summary.

Imagery pipeline: Sharp‑generated WebP + LQIP; assets referenced via typed manifest.

---

Performance Budgets

Edge/cached reads: TTFB ≤ 200 ms.

RAG reads: ≤ 2.0 s p95.

LLM gen endpoints: ≤ 10 s p95 (CPU baseline; GPU improves).

Front‑end: sub‑2s LCP, lazy‑init WebGL post‑LCP, throttle when hidden, motion respects OS prefs.

---

Telemetry & Security

Sentry for errors; PostHog for product analytics (PII redaction in place).

Auth and Secrets handled by IntegAI (Keycloak/Vault) on the API side; the website never stores secrets client‑side.

All website traffic to /api/v1/site/\* (NGINX front door → IntegAI).

---

CI/CD & Quality Gates

Vercel deploys (Preview → Production).

Lighthouse CI budget checks.

Parity soak during cutover: diff legacy vs IntegAI JSON on top slugs; require ≥99% parity for deterministic pages.

Forms idempotency test: concurrent POSTs with the same key store exactly one record.

---

Contributing

Use feature branches + PRs.

Keep section props typed; reject unknowns in the registry.

Prefer cache → local RAG → adapter fallbacks in any new API interactions.

Tag status with DECISION / SHIP / BLOCKER in PR descriptions.

---

License

Private © HISL. Reach out to the owner for reuse.

---

Appendix: Code Snippets

Typed API Client (server‑safe)

// /lib/api.ts

const API = process.env.INTEGAI\_API\_BASE!;

async function j<T>(res: Response): Promise<T> {

if (!res.ok) throw new Error(`HTTP ${res.status}`);

return res.json() as Promise<T>;

}

export type PagePayload = {

title: string;

sections: Array<{ id: string; component: string; props: any; order?: number }>;

metadata?: Record<string, any>;

lastUpdated?: string;

};

export const api = {

page: (slug: string) => j<PagePayload>(fetch(`${API}/v1/site/page?slug=${slug}`, { cache: 'no-store' })),

menu: () => j<{ primary: any[]; footer: any[] }>(fetch(`${API}/v1/site/menu`, { cache: 'force-cache' })),

seo: (slug: string) => j<any>(fetch(`${API}/v1/site/seo?slug=${slug}`, { cache: 'force-cache' })),

search: (q: string, limit = 10) =>

j<{ results: any[]; next?: string }>(fetch(`${API}/v1/site/search?q=${encodeURIComponent(q)}&limit=${limit}`)),

form: (formId: string, payload: Record<string, any>, key: string) =>

j<{ ok: boolean; idempotencyKey: string; triage?: any; route?: any }>(

fetch(`${API}/v1/site/forms/${formId}`, {

method: 'POST',

headers: { 'Content-Type': 'application/json', 'Idempotency-Key': key },

body: JSON.stringify(payload)

})

)

};

Energy Badge Utils

// /lib/energy.ts

export function estimateEnergyWh(tokens: number, baseWh = Number(process.env.NEXT\_PUBLIC\_ENERGY\_BASE\_WH ?? 0.3), per100TokWh = Number(process.env.NEXT\_PUBLIC\_ENERGY\_PER100TOK\_WH ?? 0.1)) {

return baseWh + (tokens / 100) \* per100TokWh;

}

export function whToKgCO2e(wh: number, kgPerKWh = Number(process.env.NEXT\_PUBLIC\_CO2\_KG\_PER\_KWH ?? 0.233)) {

return (wh / 1000) \* kgPerKWh;

}

R3F Globe Skeleton

// /app/globe/Globe.tsx

'use client';

import { Canvas } from '@react-three/fiber';

import { OrbitControls } from '@react-three/drei';

export default function Globe() {

return (

<Canvas camera={{ position: [0, 0, 2.6] }}>

<ambientLight />

{/\* TODO: add earth mesh, clouds, pins, packet arcs \*/}

<OrbitControls enablePan={false} />

</Canvas>

);

}

News (Grok) Server Action (pseudo‑fetch)

// /app/news/actions.ts

export async function getTrends(topic: string) {

const key = process.env.GROK\_API\_KEY;

if (!key) return { items: [], note: 'Grok disabled' };

// Pseudo‑API call: replace with the official client when enabled.

const res = await fetch('https://api.x.ai/v1/trends?q=' + encodeURIComponent(topic), {

headers: { Authorization: `Bearer ${key}` },

next: { revalidate: 30 }

});

if (!res.ok) return { items: [], error: `HTTP ${res.status}` };

const data = await res.json();

return { items: data.items ?? [] };

}

.env.example

NEXT\_PUBLIC\_SITE\_URL=http://localhost:3000

INTEGAI\_API\_BASE=https://site.hisl.ai/api

SENTRY\_DSN=

NEXT\_PUBLIC\_POSTHOG\_KEY=

NEXT\_PUBLIC\_POSTHOG\_HOST=https://eu.i.posthog.com

GROK\_API\_KEY=

DEEPSEEK\_API\_KEY=

NEXT\_PUBLIC\_ENERGY\_BASE\_WH=0.3

NEXT\_PUBLIC\_ENERGY\_PER100TOK\_WH=0.1

NEXT\_PUBLIC\_CO2\_KG\_PER\_KWH=0.233

Lighthouse CI (minimal)

// lhci.json

{

"ci": {

"collect": { "url": ["http://localhost:3000", "http://localhost:3000/globe", "http://localhost:3000/news"] },

"assert": { "assertions": { "categories:performance": ["error", { "minScore": 0.9 }] } }

}

}

---

\*\*Why this README fits your build\*\*

- \*\*Stable API surface + cutover plan:\*\* Mirrors the v1 contracts, idempotent write‑paths, and proxy→cache→takeover execution model so the website never refactors its network layer again. 5

- \*\*Performance & reliability budgets:\*\* Matches fast‑path Gemma, fallback rules, and response targets you locked, with practical frontend budgets (LCP, caching, lazy WebGL). 6

- \*\*Next.js & dev workflow:\*\* Incorporates App Router, R3F/Framer Motion, imagery pipeline, and the Cursor/Gemini local‑dev approach you documented.

- \*\*Sector depth (Pharma):\*\* Includes CROx positioning + KPIs so the site can present value, not just features. 8

If you’d like, I can also drop this into a ready‑to‑download `README.md` file.