Weeks 0–2 — lock the workflow, ship MVP behind flags

Repo guardrails (today)

Create these source-of-truth files:

component-registry.json → [{ "component": "Hero", "requiredProps": ["title","eyebrow","cta"] }, ...]

tokens.json → brand colors, spacing, radii, typography tokens (used by CI to lint props/contrast).

feature-flags.ts

export const flags = {

globe: process.env.NEXT\_PUBLIC\_FLAG\_GLOBE === '1',

news: process.env.NEXT\_PUBLIC\_FLAG\_NEWS === '1',

} as const;

component-prop-schemas/\*.json → Zod/JSON-schema per component; CI validates payloads.

Git hygiene

Enforce selective staging in Cursor (composer) and add pre-commit:

// package.json

"lint-staged": { "\*.{ts,tsx,js}": ["eslint --fix", "prettier -w"] }

Cursor “task pattern” (pin this as a snippet):

Plan changes for X. List files to edit/create, tests to add, and a Lighthouse/Axe acceptance checklist. Keep globe/news behind flags; respect tokens.json; validate props against component-registry.json.

Next.js 15.5 skeleton (Turbopack, App Router)

Minimal pages: /, /about/integai, /about/michael, /poem, /news (flagged), /globe (flagged).

Use MDX for longform; all hero/sections are block-structured with sectionId + component + props (this matches the IntegAI v1 contracts you’ll adopt later).

Imagery pipeline (no raw paths)

Put originals on NAS → run:

pnpm dlx sharp-cli ... # or scripts/process-images.mjs

pnpm generate-imagery-manifest

Always render via <Img assetId="…"> (manifest-driven). (Matches the “imagery.ts manifest + Sharp + LQIP” flow.)

Observability & privacy

Sentry (errors only), PostHog (explicit events, no autocapture). Respect DNT & cookie consent; redact free text. (Keep telemetry minimal; aligns with P1 policy.)

CI you can paste now (GitHub Actions)

name: web-ci

on: [push, pull\_request]

jobs:

check:

runs-on: ubuntu-latest

steps:

- uses: actions/checkout@v4

- uses: pnpm/action-setup@v4

- run: pnpm i --frozen-lockfile

- run: pnpm build

- name: Schema guard

run: pnpm tsx scripts/validate-props.ts

- name: Lint & typecheck

run: pnpm lint && pnpm typecheck

- name: Playwright smoke + Axe

run: pnpm e2e:ci

- name: Lighthouse CI (≥0.90)

run: pnpm lhci autorun --collect.settings.formFactor=desktop --assert.preset=lighthouse:recommended

Keep globe/news off in prod via env flags; enabled in preview to test perf & a11y before cutover.

Acceptance to exit Week 2

Pages, nav, footer, forms (idempotent) done; SEO (sitemap, canonical) wired; error/empty states covered.

CI all green; asset manifest complete; no missing alt; flags OFF in prod/ON in preview; cold-build perf meets budgets. (Maps to the locked blueprint’s “Minimal Front-End Contract” + “Acceptance Criteria”.)

Weeks 2–4 — stabilize & polish

Perf budgets: LCP ≤ 2.2 s, CLS ≤ 0.08; delay the R3F globe until after LCP and provide static fallback for reduced-motion.

Caching: edge-cache 60–120 s for news/trends with graceful last-good payload fallback.

Content ops: finalize slugs/redirects; complete alt-text; pass a11y checklist. (These guard against drift/pitfalls noted in the blueprint—slug instability, cold starts.)

Weeks 4–5 — dress rehearsal (preview flags ON)

Freeze the API v1 contracts & mock IntegAI

Stable read contracts your FE should assume permanently:

GET /v1/site/page?slug, /v1/site/menu, /v1/site/search, /v1/site/seo?slug, /v1/site/assets/{id}

Stable write contracts:

POST /v1/site/forms/{formId}, /v1/site/generate/copy, /v1/site/publish

Put thin mock adapters in your Next API routes (simulate latency/errors; idempotency via Idempotency-Key). (These are the exact surfaces IntegAI owns at cutover.)

Soak test

Flip all flags ON in preview; verify retries/backoff, fail-soft; warm caches nightly from a cache-warm.txt slug list.

Parity/k6 smoke on top 20 slugs: p95 within budget, error rate <0.5%. (Matches the phased takeover test matrix.)

Week 6 — hardware lands → IntegAI cutover (no FE refactor)

Power & rack (RAVEN path)

CEE blue socket (6 mm² spur) → PDU → Eaton 3 kVA UPS (online mode) → “RAVEN” (Threadripper + A6000).

Quick UPS runtime math you can do on day 1:

Runtime (hours) ≈ (UPS Wh × inverter efficiency × PF) ÷ load W.

Measure actual draw (wall wattmeter) under GPU load; don’t size off the 1650 W PSU label.

Host bring-up

LTS Linux, NVIDIA drivers, nvidia-container-toolkit.

Compose stack: LLM gateway (GPU), vector DB, MinIO, Postgres, Neo4j, NGINX. (This mirrors the “front door + offline RAG + artifacting + 9-core graph” locked stack.)

Data & caches

Rebuild embeddings/indexes on NVMe; warm page/search caches; shadow-test GPU endpoints.

Traffic shift (1 → 2 → 100%)

Route GETs 10–20% → observe → then POSTs → then 100%.

Keep globe/news OFF in prod until SLOs are green, then enable. (Phased Mirror→Proxy→Mutations→Decommission plan.)

Green-light gates (copy/paste into PR template)

Before hardware

✅ CI gates pass; perf/a11y budgets met; flags OFF(prod)/ON(preview); robust fallbacks render when APIs fail.

Cutover readiness

✅ nvidia-smi healthy in containers; load test p95 API <2 s, <0.5% errors; parity ≥99% for deterministic GETs; idempotent POSTs (zero dupes @ 5 RPS).

One-time Cursor helpers (ready to use)

“Wire the guardrails”

Create component-registry.json, tokens.json, and component-prop-schemas with Zod; generate a scripts/validate-props.ts that fails CI on unknown props, missing required props, or color-contrast < 4.5:1 using tokens.json.

“Add CI & budgets”

Add GitHub Actions for lint, typecheck, prop-schema guard, Playwright+Axe smoke, LHCI ≥ 0.90. Make globe/news conditional via env flags.

“Imagery pipeline”

Build Sharp pipeline + generate-imagery-manifest.mjs, convert all <Image> calls to <Img assetId="…"> with blur placeholders.

“Preview soak”

Turn all flags ON in preview; inject 2% failure at adapters; verify retries/backoff; export k6 script and LHCI report artifacts.

Why this is bullet-proof

It keeps FE stable forever by freezing the v1 API contracts now, then letting IntegAI take over behind NGINX later.

It follows your offline-first, 9-node agent graph with small, deterministic surfaces, plus artifacted releases (.integpkg) so everything is reproducible.

The cinematic style (R3F galaxy + motion + static fallbacks) is baked in without risking LCP/a11y budgets.

If you want, I can drop in the tiny mock API routes (/api/v1/site/\*) and the validate-props.ts script next.