

Project Documentation

Xai Intelligence Workspace

What this project is

This is a single-page web app built to feel like a premium, slightly futuristic “intelligence workspace.” The goal was less “build a functional dashboard” and more “make a dashboard feel alive” — heavy on motion, with a small 3D companion that roams the screen and a gradient backdrop that never quite sits still. It's built with Next.js (App Router), GSAP, Framer Motion, Three.js, and Tailwind CSS.

It's not a production product — think of it as a design/animation showcase or a starting template for a future product page, something to demo interaction ideas before they get wired up to real data.

The pieces

The AI companion

File: src/components/three/AiCompanion.jsx

This one took the most iteration. It's a small robot made entirely out of Three.js primitives — no imported models, just geometry — and it lives at the root layout, so it follows you around the entire site rather than being scoped to one section.

Its movement isn't scripted on a fixed loop. A GSAP state machine runs underneath it: every few seconds it picks a random spot on screen, works out the rotation needed to face that spot, and flies over. On arrival it does something — a scanner-beam sweep across the page, a full 360° backflip, a wave, or just an idle hover. That randomness is what keeps it feeling alive instead of like a looping GIF.

You can also grab it directly. A Raycaster runs continuously so the 3D canvas can stay pointer-events: none (meaning it never blocks clicks on the actual site) right up until the cursor is over the robot itself — at which point you can click and drag it anywhere. Release it and it picks its autonomous routine back up.

It also tracks theme changes. A MutationObserver on <html> catches light/dark toggles, reads the updated CSS variables (--accent, --surface, etc.), and applies those exact colors to the robot's materials, so it never looks out of place after a theme switch.

Background blobs

File: src/components/framer-motion/HeroBackground.jsx

A fixed, full-screen layer sitting behind everything else. A handful of large SVG gradient blobs, animated with Framer Motion, slowly morph, scale, and drift. Nothing complicated, but it's what gives the whole site its atmosphere instead of a flat background color.

Dashboard mockup

Files: DashboardPreview.jsx, InsightFlow.jsx

A fake product UI — KPI cards, a sidebar, data tables — built so the page reads as a real dashboard rather than a marketing landing page. It's mobile-first: KPI cards stack vertically on small screens, and the denser tables scroll horizontally instead of breaking the layout. Hover states and transitions are kept deliberately subtle so it reads as professional software rather than something flashy.

Hero and signature sections

Files: HeroScene.jsx, SignatureInteraction.jsx

The hero has a Three.js particle field that reacts to cursor movement, using additive blending so it glows against the gradient backdrop. Below it, a scroll-reactive section built with Framer Motion adds some parallax depth and a bit of data-viz-style motion as you scroll through it.

Tech stack

Package	Purpose
Next.js (App Router)	Framework / SPA shell
React / React DOM	UI and component architecture
Tailwind CSS v4	Styling and responsive layout
GSAP	Timeline sequencing, robot state machine, DOM animation
Framer Motion	Scroll-linked animation, background blob motion
Three.js	WebGL rendering for the companion and hero particles

Everything is plain JSX — no TypeScript in this project.

Folder layout

```
exam/ ├── PROJECT.md ├── package.json ├── jsconfig.json           (@/* -> ./src/*) ├──
next.config.mjs ├── postcss.config.mjs ├── eslint.config.mjs ├── public/ └── src/ └──
app/    ├── layout.jsx      (providers, AiCompanion, HeroBackground) └──
page.jsx    (main content flow) └── globals.css      (tokens, typography, Tailwind
config)    ├── components/ ├── theme/                (ThemeProvider, ThemeToggle)
└── gsap/      (HelloGsap) └── framer-motion/        (HeroBackground,
DashboardPreview, InsightFlow, SignatureInteraction) └── three/
(AiCompanion, HeroScene)
```

Design system

Fonts

Loaded through next/font/google in layout.jsx:

- Syne (--font-syne / font-display) — headings
- DM Sans (--font-dm-sans / font-sans) — body text
- IBM Plex Mono (--font-ibm-plex-mono / font-mono) — labels, data, code

Colors

Defined as CSS variables in globals.css. Dark mode toggles a .dark class on <html>.

Token	Light	Dark	Tailwind class
--background	#f5f5f3	#0c0c0c	bg-background
--foreground	#141414	#f2f2f0	text-foreground
--surface	ffffff	#161616	bg-surface

Token	Light	Dark	Tailwind class
--muted	#6b6b6b	#a0a0a0	text-muted
--border	#e2e2de	#2a2a2a	border-border
--accent	#0d7377	#2dd4bf	bg-accent / text-accent
--accent-foreground	ffffff	#0c0c0c	text-accent-foreground

Worth noting: the companion doesn't just approximate these colors — it parses the exact hex values (via a small `cssVarToThreeColor` helper) and applies them directly to the 3D materials, so the robot always matches the current theme exactly instead of looking slightly off.

Scripts

```
npm run dev          # dev server npm run build          # production build npm run start      # serve
the build npm run lint    # eslint
```

Notes for whoever picks this up next

- GSAP handles anything that needs real timeline control — the robot's multi-step animations especially. Framer Motion handles layout transitions and scroll-tied UI. Didn't want to force one tool to do everything it's not great at.
- Every component that touches window, document, or `requestAnimationFrame` is marked "use client" — basically all of the UI and 3D components.
- The companion respects `prefers-reduced-motion` and simply doesn't render if that's set — didn't want to force heavy motion on anyone sensitive to it.