SocialSphereMX

Source: Engagement · **Status:** In production · **Period:** 2024-03 (active)

Engagement product for creators and agencies: directory + profiles, magic-link auth, agency dashboard, social-media designer, marketplace and analytics modules, all built on a hardened VPS substrate.

The challenge

Creators and small agencies in Mexico were stitching together three or four SaaS tools to publish, measure, and monetize — and each tool took its own cut of margin plus its own data silo. The customer paid for the integration, not for the value.

My role

I designed and shipped the backend, the directory and profiles module, the agency dashboard with magic-link auth, the social-media designer, the marketplace, and the analytics module — alongside the VPS substrate that holds the whole stack together.

What I did (3–5 decisions)

1. VPS substrate first, SaaS modules second

Why: The substrate is what stops a creator from losing their data when a SaaS tool changes its terms. Building modules on top of substrate means each module can be swapped without rebuilding the foundation.

Tradeoff: More up-front substrate work than shipping modules on day one. The substrate pays for itself the first time a module has to be replaced.

2. Magic-link auth over password reset flows

Why: Passwords are a hostile surface for creators who use the platform a few times a month. Magic links cut the auth surface to one trusted channel and remove the reset-flow attack class.

Tradeoff: Magic links depend on email deliverability, so the platform needs a hardened mail path. We run our own Stalwart instance to keep the dependency out of the third-party SaaS layer.

3. Per-module evidence chain

Why: Each module's README points at a verification surface — a public advisory, a published gist, or a VPS-telemetry export. A reviewer can check the claim without taking the operator's word for it.

Tradeoff: Every module needs an artefact worth linking to. The constraint forces modules to ship with evidence instead of just shipping.

What changed

Modules in production

Baseline: 0

15 (blog CMS, agency dashboard, directory, analytics, custom domains, shop, payments, page builder, forum, AI agents, social designer, media editor, CRM, academy, calls)

Auth attack surface

Baseline: Passwords + reset flows

Magic-link only, no password DB

Third-party SaaS dependencies for substrate

Baseline: Unknown — varies by creator

One (email), self-hosted Stalwart

Overall tradeoffs

The product runs on a hardened VPS substrate rather than on a serverless platform. The cost is operational; the win is that the creator owns the substrate and can migrate modules without losing their audience.

Lessons

For engagement products the substrate IS the product. The customer is paying for control over their data, not for a glossy UI on top of someone else's serverless platform.

Stack

Next.js 16 | Prisma | SQLite | TypeScript | Tailwind | Better SQLite3 | ioredis

Repository

github.com/louzt/socialspheremx

Evidence

- [repo] github.com/louzt/socialspheremx