Hemi Product Blueprint

Last updated: 2026-03-08

1. Vision & Launch Strategy

- Mission: Deliver a credible Solana-based hub for red/black-branded crypto tooling that combines high-signal education (landing), real-time portfolio intelligence (dashboard), and a privacy-first dApp (AI concierge + anonymous transfers).
- Launch style: Fair launch (no allowlist/alpha gating). Landing, dashboard, and dApp must feel equally production-ready on day one.
- North-star experience: A user lands on Hemi, immediately understands the value prop, connects their wallet, chats with an AI assistant that actually knows their on-chain context, and can move value privately without leaving the experience.

2. Brand Voice & Narrative (English)

- Tone: Technical, confident, minimal hype. Think "we build, we ship" not "wen moon".
- Writing style: Short sentences, strong verbs, data-backed claims, optional metaphors tied to acceleration/ignition (to match the name "Hemi").
- Color & mood: Deep black background, sharp red accents (#D90429 style), optional white/gray typography for contrast. Visual references: helius.dev pacing, neon glows kept subtle.

Sample Landing Copy Blocks

1. Hero
2. Headline: "Hemi ignites Solana privacy and intelligence."
3. Subtext: "Real-time analytics, AI-native guidance, and zero-knowledge transfers—packed into one dApp."

CTA buttons: `Launch App` (primary), `Read Docs` (secondary outline).

Value Pillars

6. AI Trading Concierge: "Chat with an agent that understands on-chain flows, token risk, and your wallet activity."
7. Private Solana Rail: "Shield transfers using zk proofs so counterparties only see what you allow."

Operational Dashboard: "Monitor holdings, staking, emissions, and alerts in the same panel."

Tokenomics / Utility (if applicable): modular card layout for supply, emissions, utility.

10. Roadmap Snapshot: 3 columns (T0 launch, T1 zk upgrades, T2 cross-program automations).
11. Call to Action: "Ready for a smarter, quieter Solana stack?" + dual CTA again.

3. Experience Breakdown

3.1 Landing Page

Section	Purpose	Content Notes
Hero	Immediate clarity + CTA	Background animation (subtle red particle field), wallet connect button in nav.
Proof & numbers	Build trust	logos of infra partners (Solana, Helius, RPC, zk provider) + stats (latency, fee savings).
Utility grid	Summarize core modules	Three cards: AI Concierge, Private Rail, Dashboard.
Tech deep-dive	Show credibility	Diagram showing data flow: wallet → data layer → LLM → private tx.
Roadmap	Manage expectations	Use timeline or step cards.
Team/Backers (optional)	Social proof	Minimal silhouettes if stealth.
FAQ	Address common concerns	Gas fees, custody, auditing, supported wallets.

3.2 Dashboard

- Portfolio Overview: Holdings, PnL, staking positions (pull from Solana RPC via Helius / custom indexer).
- Alerts & Signals: Configurable rules (price delta, volume spikes, governance votes).

- AI Chat Dock: Persistent right-side panel referencing user data. Provide actions ("Explain this wallet movement", "Create transfer with shielded route").

- Transactions Feed: Blend of public + private tx (private entries display hashed metadata with reveal option).

- Settings: Wallet management, security toggles, API keys.

3.3 dApp Utility (Solana)

1. AI Chatbot
2. Sources: on-chain indexer, market APIs, internal knowledge base.
3. Capabilities: FAQ, trading guidance, generate transaction previews, trigger private transfer flows.

Stack idea: Server-side function (Edge runtime) hitting LLM (OpenAI GPT-4.1, Anthropic Claude, or self-hosted) + retrieval layer (Supabase pgvector / Pinecone) fed by Hemi docs & wallet analytics.

Anonymous Private Transfer

6. Flow: Connect wallet → choose recipient (wallet, ENS, compressed address) → select amount → generate zk proof → relayer submits transaction → user receives receipt + ability to share view key.
7. Requirements: Non-custodial, prove compliance (view keys), minimal latency.

4. Technical Architecture

4.1 Frontend

- Framework: Next.js 15 + React Server Components for landing + dashboard. Single repo with monorepo structure (apps/web, packages/ui).
- Styling: Tailwind CSS + Framer Motion for hero animations. Reusable component library (Radix UI-based) for cards, modals.
- State/Web3: Zustand or Recoil for UI state, Solana wallet adapter + wagmi-like connectors, TanStack Query for data fetching.
- Deploy: Vercel (edge functions for AI endpoints) + Cloudflare CDN for static assets.

4.2 Backend & Data

- API Layer: Next.js App Router API routes or dedicated Fastify service for Solana interactions.
- Indexer: Use Helius API (enhanced Solana RPC) for transaction parsing + webhook triggers.
- Database: Supabase (Postgres) for user profiles, preferences, chatbot conversation history, view-key metadata.
- Queues: Upstash/Kafka for AI task queueing, especially for longer analytics jobs.

4.3 AI Chatbot

- LLM Provider: Start with OpenAI GPT-4.1 or Anthropic Claude Opus, abstracted via LangChain or Vercel AI SDK for quick swaps.
- Context:
- Wallet telemetry (balances, tx history) fetched via Helius and summarized.
- Knowledge base: Markdown docs (FAQs, roadmap) stored in Supabase and embedded.
- Trading intel: Coingecko/Step Finance API + custom heuristics (volatility, liquidity).
- Safety: Rate limiting per wallet, guardrails (regex/policy) to avoid financial promises.

4.4 Private Transfer (zk on Solana)

- Candidate Protocols:
- Elusiv
  - Pros: Live on Solana mainnet, SDK friendly, supports private transfers + token swaps.
  - Cons: Liquidity pool dependency, UX may require explaining shielded accounts.
- Light Protocol
  - Pros: zk-compression, scalable data availability, integrates with Solana accounts seamlessly.
  - Cons: Still early; documentation evolving.
- Lulo / ZK Compression (custom)
  - Pros: Full control over circuits, tailor-made compliance features.
  - Cons: Longer build time, requires in-house cryptography expertise.
- Recommendation: Start with Elusiv integration for MVP (proven mainnet usage), keep Light Protocol as Phase 2 exploration for optimized fees.

- Flow Diagram:
- User wallet signs payload → client creates proof using Elusiv SDK → relayer node broadcasts → event stored in Supabase with encrypted memo + optional view key reference.

5. Delivery Roadmap (ASAP-focused)

| Phase | Duration (target) | Deliverables | |-------|------------------|--------------| | P0 Discovery | 3 days | Confirm architecture blueprint, finalize copy deck, choose zk provider, UI moodboard. | | P1 Build | 2 weeks | Landing page (responsive), Dashboard core (auth, wallet connect, portfolio, alerts), dApp shell with AI chat + placeholder private transfer flow. | | P2 Integrations | 1 week | Hook up real Solana data (Helius), connect LLM, integrate chosen zk transfer SDK, QA for mobile + desktop. | | P3 Launch Polish | 3-4 days | Security review, content proof, analytics setup, marketing assets, deployment rehearsal, fair-launch checklist. |

6. Immediate Next Steps

1. Approve this blueprint (feedback on tone/structure?).
2. Lock stack decisions (Next.js, Supabase, Elusiv, etc.).
3. Kick off design sprint: wireframes + visual system referencing helius.dev.
4. Stand up repo + CI/CD (GitHub + Vercel + preview envs).
5. Begin implementation following the phase plan above.

Let me know what to tweak; once confirmed, I'll move straight into wireframes + task breakdown. All copy/UI text will stay in English as requested.