Case Study: YieldScope

Cross-chain and exchange earning tracking

Julian M. Kleber | Builder | yieldscope.d3bu7.com

julian.kleber@sail.black | Last updated: July 22, 2026

Outcome snapshot

- Shipped a production earn ledger that syncs **Binance Simple Earn**, **OKX savings/earn**, **Monad staking**, and **LUNC** stake rewards into one dashboard — not a full portfolio or tax suite.
- Designed adapters **test-first** with golden fixtures so each source either returns real earn rows or fails closed (`ok | error | not_connected`).
- Added an onchain **EarningsCheckpoint** contract on Monad so a sync window can be posted as a portable, explorer-verifiable root hash.
- Hosted at **yieldscope.d3bu7.com** with Supabase auth, encrypted read-only CEX keys, RainbowKit wallet connect, and Kubernetes deploy on the homelab edge.

CONTEXT

Passive crypto income is fragmented. Binance Earn lives in Binance. OKX Earn lives in OKX. Monad staking rewards live onchain. Answering “what did I make this month?” still means three apps and a spreadsheet — while portfolio trackers and tax suites add balance noise instead of a sharp rewards answer.

YieldScope was built as a narrow product: sync the earn streams that matter for this personal pain, show only accrued rewards, and optionally attest a Merkle-style root on Monad so the number is portable.

PROBLEM DEFINITION

In the status quo workflow, builders who already earn on CeFi and stake on Monad face:

- no single total across exchange earn programs and onchain staking,
- portfolio dashboards that mix price swings with true reward accrual,
- tax/CSV tools that optimize for accounting exports, not a live earn checkpoint,
- DeFi browsers that ignore first-class CEX Simple Earn history.

APPROACH

YieldScope stays Phase 1 honest and depth-over-breadth:

- **Earn-native ledger** — isolate rewards/interest/staking accrual from principal price noise.
- **Four Phase 1 sources** — Binance Simple Earn, OKX savings/earn (including Auto lend and Auto Earn bill types), Monad claimed + pending staking rewards, LUNC claimed + pending stake rewards.
- **Sync windows** — import missing since last sync (default), re-download full history, or custom UTC date range; auto-import on open when history already exists.
- **Fail-closed adapters** — broken or missing credentials never invent earn rows.

- **Onchain attestation** — hash a sync window and post it via `EarningsCheckpoint` on Monad mainnet (attest remains disabled until the contract address is configured).

SYSTEM OUTLINE

- **Auth**: Supabase email/password with confirmation; session-gated `/app/*` and sync/attest APIs.
- **CEX connect**: read-only Binance / OKX API keys encrypted server-side per user.
- **Wallet**: RainbowKit + wagmi on Monad mainnet (chain id 143) for stake reads and attestation; Phantom-focused connect UX.
- **LUNC**: paste Terra Classic address for stake reward history + pending via FCD/LCD.
- **Adapters**: typed sync modules with unit tests against golden fixtures; live smoke gated by secrets.
- **Checkpoint**: Foundry `EarningsCheckpoint.sol`; explorer-verifiable root for a sync window.
- **Ops**: Next.js app, Supabase ledger schema, k8s manifests, nginx edge TLS for `yieldscope.d3bu7.com`.

IMPLEMENTATION STACK

- **Frontend / API**: Next.js App Router, RainbowKit / wagmi.
- **Data / auth**: Supabase (Postgres + Auth); SQL migrations for the earn ledger.
- **Contracts**: Solidity + Foundry (`EarningsCheckpoint`).
- **Chain**: Monad mainnet RPC for stake reads and attestation; optional explorer API for claim history.
- **Quality**: Vitest adapter tests, Foundry contract tests, Playwright auth E2E, live smoke script.
- **Deploy**: Docker image, Kubernetes on blackpearl/engine, edge nginx for public TLS.

RESULTS (HONEST PHASE 1)

- **Product live** at yieldscope.d3bu7.com with docs for connecting wallets and exchanges.
- **Source coverage locked** to Binance, OKX, Monad, and LUNC — Phase 2 multi-chain DeFi aggregators explicitly deferred.
- **Test-proven adapters** for each Phase 1 source; dashboard statuses stay fail-closed.
- **Public repo** — cap-jmk-launchpad/YieldScope (Spark / BuildAnything submission track).
- **Checkpoint discipline** — attest UI stays disabled until a mainnet contract address is set; no invented addresses.

No fabricated user-count or APY metrics: YieldScope's claim is a working earn ledger and verifiable checkpoint path, not portfolio AUM.

WHAT THIS CASE STUDY DEMONSTRATES

- Blockchain product sense: CeFi earn + L1 staking in one pane, with an onchain proof layer.
- Reliability-first integration engineering — adapters that fail closed under real exchange and RPC quirks.
- Full-stack delivery from brand/landing through contracts, auth, sync, and homelab production hosting.
- Honest scoping: depth on Phase 1 sources instead of claiming Zerion-breadth DeFi coverage.

WHAT I WOULD IMPROVE NEXT

- Deploy and verify `EarningsCheckpoint` on Monad mainnet; enable attest end-to-end.
- Harden claim-history paths against public RPC log-range limits without requiring a self-hosted archive.
- Expand Phase 2 sources (ETH / Lido / Base) only after Phase 1 DoD stays green under live keys.