



## SDG BLOCKCHAIN ACCELERATOR

### Technical Architecture Document – KarbonLedger

## 1. Project Information

- **Project Name:** Streamline — Powered by KarbonLedger
- **Challenge & UNDP Office:** Effluent Treatment Plant (ETP) Transparency & Climate Credits — UNDP India
- **Document Version:** v1.0

## 2. Overview

This prototype implements a **real-time compliance and climate accounting system** for Common Effluent Treatment Plants (CETPs).

- CETP sensor data is ingested, validated with AI and hashed on the **Cardano blockchain**.
- **Aiken smart contracts** manage the lifecycle of **Carbon Emission Tokens (CETs)** and **Carbon Offset Tokens (COTs)**.
- The **Streamline dashboard** (Next.js + Supabase) provides operators and regulators a trusted interface for compliance reporting, with SDG-aligned metadata.

## 3. System Architecture Diagram

```
[Frontend (Next.js UI)]
  ↓ Project registration / validator voting
[Supabase DB & APIs]
  ↓ On approval event
[Smart Contracts (Aiken)]
  ↓ Mint/Burn events
[Cardano Blockchain (via Blockfrost)]
  ↓ Webhook parsing
[Supabase Carbon Tables & Registry]
  ↓
[Streamline Dashboard: Operator / Regulator / Public View]
```

## 4. Blockchain Design

- **Network:** Cardano blockchain (Preview testnet, via Blockfrost API).
- **Reason for choice:** Low transaction costs, energy-efficient consensus and strong support for UTXO-based compliance logic.
- **Ledger function:** Immutable anchoring of CETP data → Carbon Emission Tokens (CETs) and Carbon Offset Tokens (COTs).
- **Data anchoring:**
  - **Off-chain:** Project metadata, validator votes and balances in **Supabase (Postgres)**.
  - **On-chain:** Token mint, burn and transfer events with **SDG-linked metadata**.
- **Token Model:**
  - **COT (Carbon Offset Token):** Minted after project approval; transferable.
  - **CET (Carbon Emission Token):** Burned against COTs; represents emissions liability.

## 5. Data Flow & Transaction Lifecycle

1. **Registration:** User submits CETP project → stored in Supabase (**projects**).
2. **Verification:** Validators cast votes → logged in Supabase (**votes**).
3. **Approval:** Smart contract enforces  $\geq 3$  approvals → triggers COT mint.
4. **Minting/Transfers/Burns:** Detected via Blockfrost webhook → **carbon\_offset\_actions** updated.
5. **Balances:** Aggregated in views (**carbon\_offset\_balances**, **carbon\_offset\_token\_status**).
6. **Registry & Reporting:** Public dashboard shows balances, status, and SDG-linked impact

## 6. Off-chain Components

### Backend services

- **Supabase (Postgres + Auth):**  
Used as the primary backend for storing project submissions, validator votes, token lifecycle events (**carbon\_offset\_tokens**, **carbon\_offset\_actions**) and balances.
- **Next.js API Routes:**  
Minimal API endpoints for project registration, validator voting and token status queries.
- **Webhook Handler:**  
Blockfrost webhook integration is live — listens for mint/transfer/burn events and

updates Supabase tables accordingly.

## Integration with Cardano node / testnet

- **Network:** Cardano Preview Testnet.
- **Access Method:** Via **Blockfrost API & Webhooks** (no local node yet).
- **Smart Contracts:** Aiken/Plutus validators for project verification and token logic have been tested in the sprint environment.

## Oracles / data providers

- **Data Source (Current):**  
CETP operational and test data is **manually ingested or simulated** during prototype stage.
- **Planned Integration:** IoT sensor streams for flow, COD, BOD, TDS, pH, energy — not yet wired but scoped for pilot.

## Dashboard or reporting tools

- **Streamline Dashboard (Next.js + Tailwind + shadcn/ui):**
  - Displays project submissions, verification status and token lifecycle (minted, transferred, retired).
  - Pulls directly from Supabase views (`carbon_offset_balances`, `carbon_offset_token_status`).
- **Admin View:** Provides visibility of validator votes and approval workflows.

## 7. Sandbox/Testnet Results

- **Network:** Cardano Preview Testnet (via Blockfrost)
- **Smart Contract Testing:** Aiken CLI unit tests + emulator tests
- **Integration:** Successful webhook integration — tested mint, transfer, burn detection flows
- **Outputs:**
  - Verified transaction anchoring
  - Validator rules enforced

- **Token lifecycle visible in Supabase-backed dashboard**

## 8. Tools and Environments Used

### Cardano-node

- Not yet run locally in the sprint; blockchain access provided via **Blockfrost API** connected to the **Cardano Preview Testnet**.
- Local node + Ogmios/Kupo integration planned for pilot stage.

### Aiken CLI

- **Version:** v1.1.17
- Used for compiling and checking smart contracts (`aiken build`, `aiken check`).
- Contracts include project validator, token minting logic, and multisig approval flow.

### Ogmios / Kupo

- **Not used in sprint phase.** Prototype relies on Blockfrost webhooks for transaction monitoring.
- Ogmios/Kupo considered for future integration to enable low-latency local UTXO indexing and richer chain queries.

### Testnet / Emulator

- **Cardano Preview Testnet** used for all contract deployments and mint/transfer/burn testing.
- Emulator tests run through **Aiken CLI** for validating validator logic and edge cases.

## 9. Remaining Considerations / Next Steps

- **Pending Architecture Changes**

- Transition from **Blockfrost-only integration** to a hybrid setup with **cardano-node + Ogmios/Kupo** for richer chain queries and lower latency.
- Full integration of **IoT data ingestion service** (flow, COD, BOD, energy) into Supabase and linking it to on-chain compliance records.

- **Performance Improvements**

- Optimize **Aiken validator scripts** to reduce memory usage and transaction costs.
- Enhance **event parsing logic** in webhook handler to handle higher throughput and batch transactions efficiently.
- Implement caching and async queues for real-time dashboard updates.

- **Security Audit Steps**

- Conduct external **audit of Plutus/Aiken validators** to verify correctness and safety (e.g., project approval and token minting logic).
- Review **RBAC and Supabase RLS policies** to ensure data integrity and prevent unauthorized writes.
- Add **end-to-end encryption** for IoT data pipelines once sensors are integrated.

- **Next Deployment Milestones**

- Deploy contracts and dashboard on **Cardano Preprod Testnet** for extended stress testing.
- Expand stakeholder testing sessions with SPCB officials and CETP operators to validate reporting formats and workflows.
- Document and publish **pilot deployment plan** for one CETP site with full IoT–AI–Blockchain integration.