




## SDG BLOCKCHAIN ACCELERATOR

# Debugging and Testing Report – zenGate Global Ltd

## 1. Project Information

- **Project Name:** zenGate Global - Traceability via Winter Protocol
- **Challenge & UNDP Office:** UNDP Bangladesh
- **Document Version:** v1.0

## 2. Testing Approach

While TraceIT leverages existing Winter Protocol smart contracts (already reviewed and tested prior to the Accelerator), the primary work during the Accelerator focused on building the **application layer** on top of these contracts.

Accordingly, most debugging and testing was performed at the **API, frontend, and integration layers**, rather than writing new Aiken validators.

- **Unit Testing (Smart Contracts):**
    - Core validator logic was not materially changed during this PoC.
    - Existing Aiken contracts already covered with unit tests (redeemer schema checks, invalid datum rejection, double-spend prevention).
    - We did not add new unit tests in this phase.
- **Integration Testing (Application Layer):**
    - Conducted repeated end-to-end tests against the **Cardano Preview Testnet** using wallets (Gero, Eternl) to simulate real user flows.
    - Verified record creation, tokenization via Winter Protocol, IPFS uploads, and on-chain anchoring. You can view all of these tests in the application explorer.
    - Database integration tested with Postgres (record insertion, linkage to Winter IDs, wallet verification).
- **Application Edge Cases Tested:**
    - Wallet connection interrupted or user cancels signing.

- Wallet/network mismatch (Mainnet vs Preview).
- Invalid or expired nonce during wallet verification.
- API payloads with missing/invalid fields.
- Preview tokenization simulated without persisting DB record.

## 3. Error Logs

*N/A -no significant smart contract tests conducted*.

### Application Debugging Workflow

- **Test Network:**
  Used **Cardano Preview Testnet** for end-to-end testing. Ensured wallets were set to Preview and environment configs pointed to Preview services (e.g., `PREVIEW_WINTER_URL`, `BLOCKFROST_PROJECT_ID`).

- **Browser-first Debugging:**
  Since TraceIT is a browser-based application, most debugging was performed using **Developer Tools** and console logs:
  - Inspected requests to `/api/**` in the Network tab to check payloads and JSON responses (`success`, `message`, `error`, `code`).
  - Checked client state in `localStorage`: `wallet-verification-storage` and `selectedNetwork` keys influenced UX; clearing them helped reset sessions.
  - Monitored wallet errors in console (e.g., `DataSignError`, user-cancelled signing).

## 4. Resolved Issues

**PoC shortcuts and known gaps (still open):**

- Signature verification on the server is minimal (structure check); full cryptographic verification should be implemented.
- Error handling is user-friendly but not exhaustive.
- Migrations are optional; current flow encourages direct schema push for speed.

## 5. Optimization Notes

- *Main optimization notes are around transaction cadence and strategy. We have created a workflow to ensure that we optimize around cost per event.*

## 6. Tools and Environments Used

- Aiken CLI v1.0.26-alpha+075668b (`aiken check`, `aiken build`)
- Cardano-node 8.9.0 (Preview Testnet)
- Mesh SDK: transaction building, wallet interaction (React hooks), UTXO selection.
- Blockfrost: network data provider for Mesh transaction builder for Cardano payments.
- Winter Protocol: To create records on chain via Winter API.
- IPFS gateways: public viewers (e.g., `ipfs.io`) for content addressing.

## 7. Remaining Issues / Next Steps

- Pending improvement: validator script size and execution cost reduction
- Next step: integrate private IPFS gateway and Winter v2 Smart Contract upgrades