SDG BLOCKCHAIN ACCELERATOR

Prototype (PoC) 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. Project Overview

The **Winter Protocol API Service Pilot** addresses a critical adoption barrier in Bangladesh's traceability sector: existing platforms such as **SERA** are cloud-first and not blockchain-native, which limits trust, transparency, and long-term interoperability.

This project implements a **plug-and-play blockchain gateway** for traceability platforms, enabling them to **anchor verified events on Cardano** via APIs and JSON uploads, without requiring blockchain expertise. By abstracting wallet, key management, and transaction complexity, the service makes blockchain integration accessible to non-technical platform developers, UNDP staff, and other stakeholders.

3. Repository Structure

Core Codebases

The traceability protocol is implemented across three open-sourced repositories:

1. **Winter-Cardano-Contracts**
   Aiken-based Plutus V2 smart contracts for Cardano. Provides the on-chain logic for creating, updating, and retiring stateful traceability events.
   Repo: https://github.com/zenGate-Global/winter-cardano-contracts

2. **Winter-Cardano (Library)**
   A TypeScript/JavaScript library that compiles and consumes the contracts. Exposes safe builders to instantiate events, pin metadata to IPFS, mint singleton tokens, recreate/update event state, and spend/retire events.
   Repo: https://github.com/zenGate-Global/winter-cardano/tree/main

3. **Winter-Backend (Cardano Service)**
   A backend service (PoC/MVP) that wraps the library into APIs. Handles IPFS pinning, transaction building, signing, submission, and indexing into a SQL DB. Includes retry

logic, monitoring, and integration guides.
Repo: https://github.com/zenGate-Global/winter-backend-cardano

4. Build Instructions

1. **Winter-Cardano-Contracts -** please view ***readme*** on the first page of the repository - https://github.com/zenGate-Global/winter-cardano-contracts

2. **Winter-Cardano (Library) -** view ***readme*** on the first page of the repository - https://github.com/zenGate-Global/winter-cardano/tree/main

Prerequisite:

- You need to have the Aiken compiler installed and configured on your system. This project is configured to use Aiken v1.0.26-alpha.
- Note that building is only needed if the contract source code is modified or a different compiler version is used. Otherwise, the build output, plutus.json, can simply be copied.

```
# Compile Aiken contracts
aiken build
```

5. Test Instructions

(Explain how to run unit and integration tests.)

```
# run tests
aiken check
```

6. Deployment Instructions

*Please **follow detailed instructions here -***
https://github.com/zenGate-Global/winter-cardano/tree/main?tab=readme-ov-file

```
# Set up Event Builder
// Mainnet
const winterEvent = new EventFactory('Mainnet');

// Preview
const winterEvent = new EventFactory('Preview');

// Preprod
const winterEvent = new EventFactory('Preprod');

// Custom
const winterEvent = new EventFactory('Custom');

# Set up Provider - e.g BlockFrost

const provider = new Blockfrost('<apiUrl>', '<projectId>');
await winterEvent.setProvider(provider, {
  seed: '<seedPhrase>'
});

# Set up Contract

const walletAddress = await winterEvent.getWalletAddress();
const walletAddressPK = winterEvent.getAddressPK(walletAddress);

await winterEvent.setObjectContract({
  protocolVersion: 1n,
  dataReference: fromText('harvest'),
  eventCreationInfo: fromText(''),
  signers: [walletAddressPK]
});


# Minting

const name = 'testSingletonTracker';
const walletUtxos = await winterEvent.getWalletUtxos();
```

```
const completeTx = await winterEvent.mintSingleton(name, walletUtxos);
const signedTx = await completeTx.sign().complete();
const txHash = await signedTx.submit();
await winterEvent.waitForTx(txHash); // optional confirmation

# Build Recreaction Transaction

const newDataRef = ["deadbeef"];
const completeTx = await winterEvent.recreate(walletAddress, utxos,
newDataRef);
const signedTx = await completeTx.sign().complete();
const txHash = await signedTx.submit();
await winterEvent.waitForTx(txHash);

# Build Spending Transaction - Koios example

const completeTx = await winterEvent.spend(
  walletAddress,
  walletAddress,
  utxos,
  '<koiosUrl>'
);
const signedTx = await completeTx.sign().complete();
const txHash = await signedTx.submit();
await winterEvent.waitForTx(txHash);
```

Contract Fields Explained

- protocolVersion: Which version of the protocol is being used (bigint).
- dataReference: Arbitrary data in hex (e.g. IPFS link).
- eventCreationInfo: The original txId. Typically an empty string at creation.
- signers: Array of payment credential hashes of authorized signers.

7. Testnet Results

(Provide evidence of successful deployment and contract execution.)

Transaction ID	Contract Action	Status	CPU	Memory	Notes
https://preprod.cardanoscan.io/transaction/2156e6f3c7c25446903771f6bd45c82d06437de174aabba0571133a1a4c4e013	Mint Event UTxO	Success	3000000000	7000000	Validator executed correctly
https://preprod.cardanoscan.io/transaction/2dea04eb4886ccd576a2d591a1d1c9982ec7fa27241ae2db55f9191a60f62194	Recreate Event UTxO	Success	3000000000	7000000	Validator executed correctly
https://preprod.cardanoscan.io/transaction/7868e4be9f340caa26887783b464bef8bbb07eb9476f0fe7a95e580d3ffee261	Spend Event UTxO	Success	3000000000	7000000	Validator executed correctly

8. Dependencies & Environment

- **Prerequisites:**
  - Have your seed phrase prepared with some ADA in the wallet.
  - Have your provider details ready.

- You need to have the Aiken compiler installed and configured on your system. This project is configured to use Aiken v1.0.26-alpha.

9. Demo / Walkthrough

- Video Link of Prototype walk through: https://drive.google.com/file/d/1RmAh5IED-Qqjc3AndzZbULnf2ethCYkj/view?usp=sharing

10. Remaining Issues / Next Steps

Internal Review and Next Steps

- **Review and Feedback:** https://docs.google.com/document/d/13p2rcdRmLc7QOlvvN_q6CDgqQUt3fLrDH272QAHHPz8/edit?tab=t.0#heading=h.xb1h97u931c1