SDG BLOCKCHAIN ACCELERATOR

Prototype (PoC) Report – Template

1. Project Information

- **Project Name: BlackFrog**

- **Challenge & UNDP Office:**
  Rising ethnic tensions in the Balkans and Central Asia are fueled by misinformation and untrusted systems, alongside opaque mineral supply chains that hinder economic independence. **UNDP Istanbul Regional Hub**

- **Document Version:** v1.0

2. Project Overview

This prototype showcases a tokenized RWA investing platform for pre-financing critical raw materials on Cardano. Users can subscribe to STOs, manage tokenized assets, and track yield distribution, with all transactions executed and validated on-chain. **(name features of current POC)**

Key features:

- Contributors can fund projects by sending ADA (will be replaced by a stablecoin in production) to the smart contract.
- Automatic validation of contribution rules (minimum contribution, fundraising deadlines, target caps).
- Refund mechanism if the target is not reached.
- Owner withdrawal and distribution functionality once the target is reached.
- Off-chain scripts (Node.js with Lucid Evolution + Blockfrost) handle contract initialization, contributions, and refunds.

3. Repository Structure

Here is the repository link : https://github.com/Elizaproai/rwa-cardano-launchpad

The structure is as follow :

```
crowdfunding/
├── aiken.toml
├── validators/
│   └── fundraising.ak     → Aiken smart contract source code
├── plutus.json            → Compiled Plutus V3 validator
├── scripts/               → Node.js scripts for initialization,
│    ├── 1-initialize.js      contribution, refunds, withdrawals
│    ├── 2-contribute.js
│    ├── 3-refund.js
│    ├── 4-withdraw.js
│    └── utils.js
├── .env.example     → The sample .env file
└── README.md        → The Readme file
```

4. Build Instructions

```
# Compile Aiken contracts
cd crowdfunding
aiken build -t verbose

Aiken version : aiken v1.1.17+c3a7fba

2. Run scripts step by step
Install dependencies
cd rwa-cardano-launchpad/scripts
npm install
Setup environment variables
Copy and edit your .env file:

cp env.sample .env
Fill in the variables:

BLOCKFROST_API_KEY: Get one from Blockfrost
BLOCKFROST_URL: Blockfrost endpoint (e.g.,
                                     https://cardano-preprod.blockfrost.io/api/v0)
NETWORK: Cardano network (e.g., Preprod)
ADMIN_ADDRESS: Your admin wallet address
ADMIN_MNEMONIC: Mnemonic phrase for admin wallet
USER_MNEMONIC: Mnemonic phrase for a contributor wallet
```

5. Test Instructions

Unit tests are not yet written.

For the integration test, you have it down here :

```
# Run scripts step-by-step:
cd rwa-cardano-launchpad/scripts

node 1-initialize.js   # Deploy contract with initial datum
  Example output:
  Contract Address:
addr_test1wrdzsldt65nfpsy3e7gzfmfhstuwm3wjz5qxyt8e5jn0jegzlz7z2
  txHash:
306f36a5332040da394a1652ef792b9214f92b60fc46e201a9f33d176a0d1245

node 2-contribute.js   # User contributes funds

node 3-refund.js       # Contributor refunds if goal not met

node 4-withdraw.js     # Owner withdraws if goal is met

# Edge cases covered:
Contribution below 1 ADA rejected
Contribution after deadline rejected
Withdrawal before deadline rejected
Refund attempt when target met rejected
```

6. Deployment Instructions

```
# Deploy using Node.js scripts with Lucid Evolution:

   -  Initialize contract with 1-initialize.js (sets target amount, dates,
      interest, etc.) :
      node 1-initialize.js

   -  Fund contract with 2-contribute.js :
      node 2-contribute.js

   -  Run 3-refund.js (if campaign fails) :
     node 3-refund.js

   -  Run 4-withdraw.js (if campaign succeeds) :
      node 4-withdraw.js

# Prerequisites:
   -  Funded testnet wallet (Preprod ADA via Cardano faucet).
   -  Blockfrost account + API key.
   -  Users MNEMONIC to fill in the .env file
          -  ADMIN MNEMONIC
          -  USER1 MNEMONIC
          -  USER2 MNEMONIC
```

7. Testnet / Emulator Results

Transaction ID	Contract Action	Status	CPU	Memory	Notes
0957afd25c049c11fe3f37bf1e1af37a1e9083bf54abbdff8e130e1aefe0fd1e	Initialize	Success	-	-	Datum set with target = 5 ADA Script duration : 0m1.406s
6a535f092c026f9e00089d89def38da94aaf3c1f25ba8f833d	Contribution	Success	-	-	1 ADA contribution validated Script duration : 0m1.609s

`976a9e6de0c7cb`					
`efa52516e171f05c8948c1a45b2805ac3f3d63ef3a8cbfe5dc069f12daccf7d8`	Refund	Success	-	-	Contributor refunded 1 ADA Script duration : 0m1.623s
`0cadff5764abb1486c50736aa7d7e33279188337b572b52863db06264d6cb21d`	Withdraw	Success	-	-	Owner withdrew funds after deadline. Script duration : 0m1.584s

(Provide evidence of successful deployment and contract execution.)

- Include screenshots of dashboard or transaction confirmations if available.

8. Dependencies & Environment

- Aiken CLI : v1.1.17+c3a7fba
- Lucid Evolution: ^0.4.29
- Blockfrost SDK: ^6.0.0
- Node.js: v23+
- Network: Preprod Testnet
- Testnet used (Preprod)
- Any libraries, SDKs, or off-chain services

9. Demo / Walkthrough

Here is the video link demo for **user investing in an ongoing fundraising** feature :

https://drive.google.com/file/d/1yIv_J9b58L0LAstu3tw2viAH2i3TAxH7/view?usp=sharing

10. Remaining Issues / Next Steps

- Add unit tests for implemented validators (contribute, claimRWA, refund, ownerWithdraw)..
- Optimize validator to minimize script execution units.
- Deploy and use RWA token.
- Use a stable coin for contributions, instead of ADA.
- Implement user claim RWA token on the frontend.
- Implement user Burn RWA token on the contract and frontend.
- Final deployment to Cardano Preprod and then Mainnet after review.