SDG BLOCKCHAIN ACCELERATOR

Technical Architecture Document – 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**

- **Report Version:** v1.0

2. Overview

This project implements a tokenized RWA (Real World Assets) investing platform on Cardano using Aiken smart contracts.

The prototype allows pre-financing of critical raw materials by enabling users to participate in tokenized fundraising campaigns (STOs).

Problem it solves:

- Lack of transparent and trusted funding mechanisms in resource supply chains.
- High reliance on intermediaries for verification and fund distribution.
- Limited investor confidence due to absence of automated refund/withdraw mechanisms.

Purpose of PoC:

- Demonstrate a decentralized crowdfunding mechanism with automatic rules validation.
- Showcase refund and withdrawal flows on-chain.
- Serve as a foundation for future integration of stablecoins and RWA token issuance.

3. System Architecture Diagram

```
User Wallets  --->  Transaction  --->  Validator Script  --->  Cardano Ledger (UTxO)

    |                                ^

    v                                |

 Off-chain Services (Lucid Evolution + Blockfrost) --+
```

Components to include:

- User Wallets: Admin + contributors interacting via dApp.
- Aiken Validator Scripts: Plutus V3 contract written in Aiken.
- Off-chain Components: Node.js scripts with Lucid Evolution + Blockfrost API.
- Cardano Ledger (UTxO): Tracks contract state, contributions, and outcomes.
- External Data Sources: None in current PoC (future: stablecoin price oracles).

4. Blockchain Design

- **Smart Contracts:**
  - **Fundraising Validator (Plutus V3 via Aiken):**
    - Validates contributions.
    - Enforces refund if target not reached.
    - Allows owner withdrawal when fundraising is successful.
- **Datum Structure:**
  - ownerPkh (contract owner).
  - startDate, endDate.
  - interestRate.
  - targetAmount.
  - currentRaised.
  - contributors[] (list of contributor PKHs + amounts).
- **Redeemer Structure :**
  - Contribute(pkh, amount, timestamp)
  - Refund(pkh)
  - Withdraw()
- **UTxO Model Usage :**
  - Contract state stored in UTxO with inline datum.
  - Contributions consume & update the datum UTxO.
  - Refunds/withdrawals consume UTxO and reallocate ADA.
- **Token Management (future work) :**
  - RWA tokens to be minted when the campaign succeeds.
  - Tokens burned upon redemption.

- **Security Considerations :**
  - Signature checks: contributor PKH for refunds; admin PKH for withdrawal.
  - Datum validation: ensures consistency in state transitions.
  - Deadline checks: prevents early withdrawal or late contributions.
  - Replay protection: ensured by consuming and recreating the state UTxO.

5. Data Flow & Transaction Lifecycle

1. **Initialize**: Admin deploys contract with datum (fundraising params).
2. **Contribute**:
   a. User submits ADA → validator checks contribution rules.
   b. Datum updated with new contributor and raised amount.
3. **Refund** (if failed):
   a. After the deadline, the contributor requests a refund.
   b. Validator verifies target not met and contributor’s balance.
   c. Funds returned.
4. **Withdraw** (if successful):
   a. Admin requests withdrawal.
   b. Validator checks fundraising success.
   c. Funds released to admin.
5. **Off-chain updates**: Lucid scripts update user dashboards with transaction results.

6. Off-chain Components

- Backend services: Node.js scripts (initialize, contribute, refund, withdraw).
- Integration: Lucid Evolution for transaction building, Blockfrost for chain queries.
- Dashboards: Future frontend app planned for user interaction.
- No oracles yet: Stablecoin oracle will be added in production.

7. Sandbox/Testnet Results

Transaction ID	Type	Status	CPU	Memory	Notes
0957afd25c049c11fe3f37bf1e1af37a1e9083bf54abbdff8e130e1aefe0fd1e	Initialize	Success	-	-	Datum set with target = 5 ADA Script duration : 0m1.406s
6a535f092c026f9e00089d89def38da94aaf3c1f25ba8f833d976a9e6de0c7cb	Contribution	Success	-	-	1 ADA contribution validated Script duration : 0m1.609s
efa52516e171f05c8948c1a45b2805ac3f3d63ef3a8cbfe5dc069f12daccf7d8	Refund	Success	-	-	Contributor refunded 1 ADA Script duration : 0m1.623s

0cadff5764abb1486c50736aa7d7e33279188337b572b52863db06264d6cb21d	Withdraw	Success	-	-	The owner withdrew funds after the deadline. Script duration : 0m1.584s

8. Tools and Environments Used

- Aiken CLI: v1.1.17+c3a7fba (build, check)
- Lucid Evolution: ^0.4.29 (transaction builder)
- Blockfrost SDK: ^6.0.0 (blockchain queries)
- Node.js: v23+
- Network: Cardano Preprod Testnet

9. Remaining Considerations / Next Steps

- Add unit tests for validators.
- Optimize validator to reduce execution units.
- Deploy RWA token minting & burning policy.
- Switch contributions from ADA → stablecoin.
- Add frontend support for claim RWA tokens.
- Perform stress tests on Preprod before Mainnet deployment.
- Plan for security audit (validator & off-chain).