SDG BLOCKCHAIN ACCELERATOR

Prototype (PoC) Report – Template

1. Project Information

- **Project Name:** Blockchain-Enabled CRF Fund Disbursement System
- **Challenge & UNDP Office:** UNDP Bangladesh
- **Document Version:** V1

2. Project Overview

This prototype demonstrates a **hybrid cooperative loan management platform** that integrates **Cardano blockchain smart contracts** with **traditional finance systems** (mobile money/ fiat).

The solution enables:

- **Primary cooperative admins** → onboard beneficiaries, set loan terms, track repayments.
- **Secondary cooperative committees** → approve or review loan requests.
- **Beneficiaries** → apply for loans, receive disbursements (on-chain + fiat), and make repayments via mobile money.
- **Blockchain layer** → provides **transparency, immutability, and auditability** of loan agreements and repayments.
- **Fiat/MFS integration** → ensures end-users interact via familiar mobile wallets, while the cooperative backend syncs with blockchain records.

Main features:

- On-chain **loan contracts** and cooperative governance (Aiken validators).
- Off-chain **admin dashboard** for loan & user management.
- **Mobile money integration** for seamless disbursement and repayment.
- Role-based access: **Admin, Committee, Beneficiary**.
- DID support for **secure digital identities**.

3. Repository Structure

(Outline how your code and related files are organized.)

Suggested Structure:

```
/crf_app
```

```
    /app
        /templates        → HTML frontend (loan dashboard, forms, status
pages)
        /blockchain_integration → Cardano/Plutus integration helpers
        /loan_manager.py   → Loan creation, disbursement, repayment logic
        /models.py         → Off-chain data models (members, loans,
repayments)
        /main.py           → Flask app entry point (API endpoints)
/deployed_contracts       → Conceptual Plutus scripts and generated
addresses
/scripts                  → Transaction build/sign/submit scripts
/docs                     → README.md, deployment guide, architecture
notes
```

4. Build Instructions

```
# Install dependencies
pip install -r requirements.txt

# Set environment variables
export BLOCKFROST_PROJECT_ID=<your_blockfrost_key>
export NETWORK=preprod

# Run Flask backend
python main.py
```

5. Test Instructions

Ensure your virtual environment is activated. Then, run the main application file:

```
    ```bash
    python app/main.py
    ```
```

You should see output indicating the Flask server is running, typically on `http://127.0.0.1:5000/`.

Access the Frontend:

Open your web browser and navigate to `http://127.0.0.1:5000/`.

Expected outputs:

- Successful DID registration → returns DID string.
- Loan approval by committee → triggers on-chain contract validation.
- Repayment logging → hash stored on-chain, verified in DB

6. Deployment Instructions

Deploy contracts to Cardano testnet:

```
# Build transaction
cardano-cli transaction build \
  --testnet-magic 1 \
  --tx-in <UTXO> \
  --tx-out <ADDRESS>+1000000 \
  --change-address <CHANGE_ADDR> \
  --out-file tx.raw

# Sign transaction
cardano-cli transaction sign \
  --tx-body-file tx.raw \
  --signing-key-file cooperative.skey \
  --testnet-magic 1 \
  --out-file tx.signed

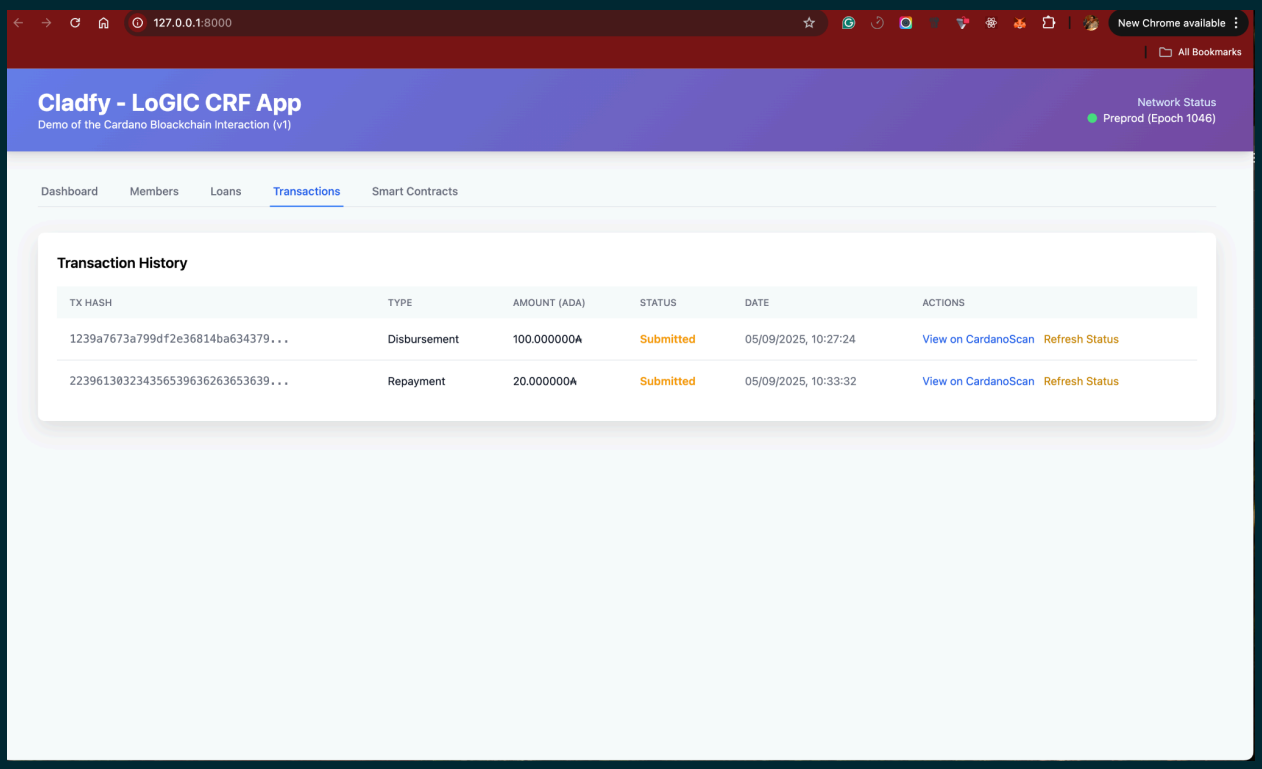
# Submit transaction
cardano-cli transaction submit \
  --testnet-magic 1 \
  --tx-file tx.signed
```

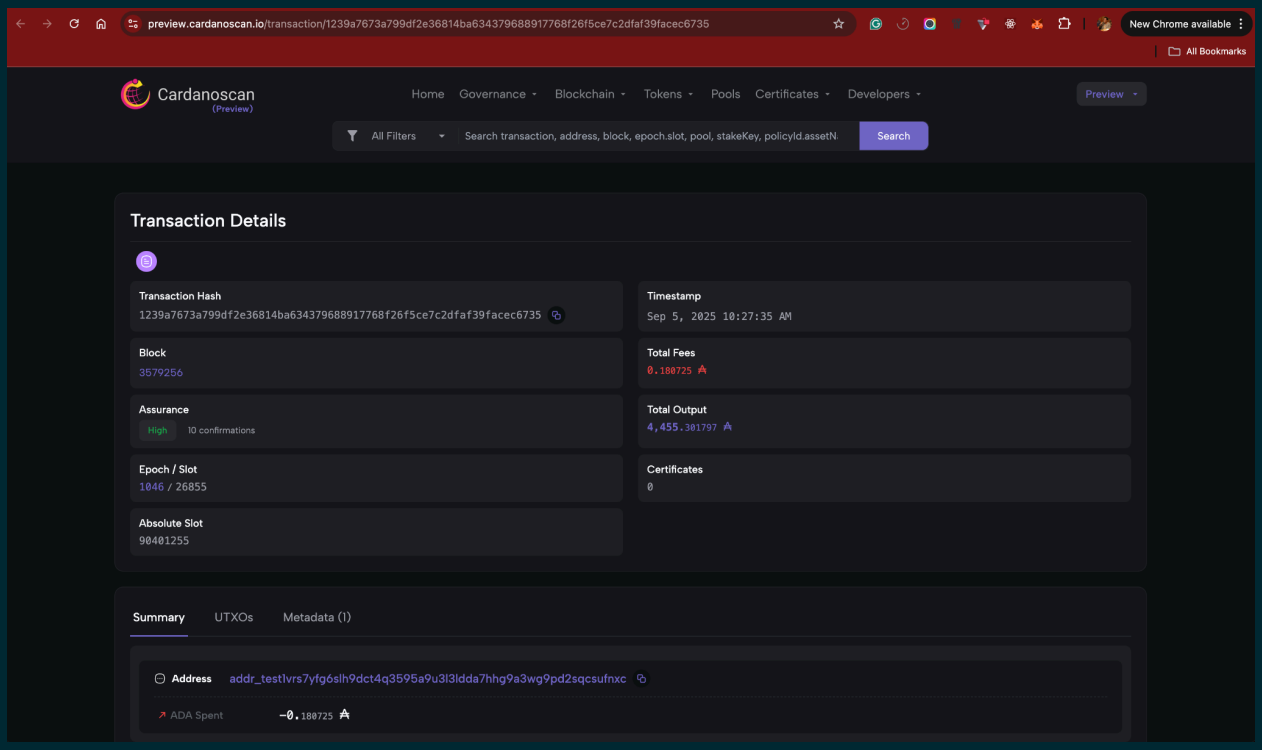
Prerequisites:

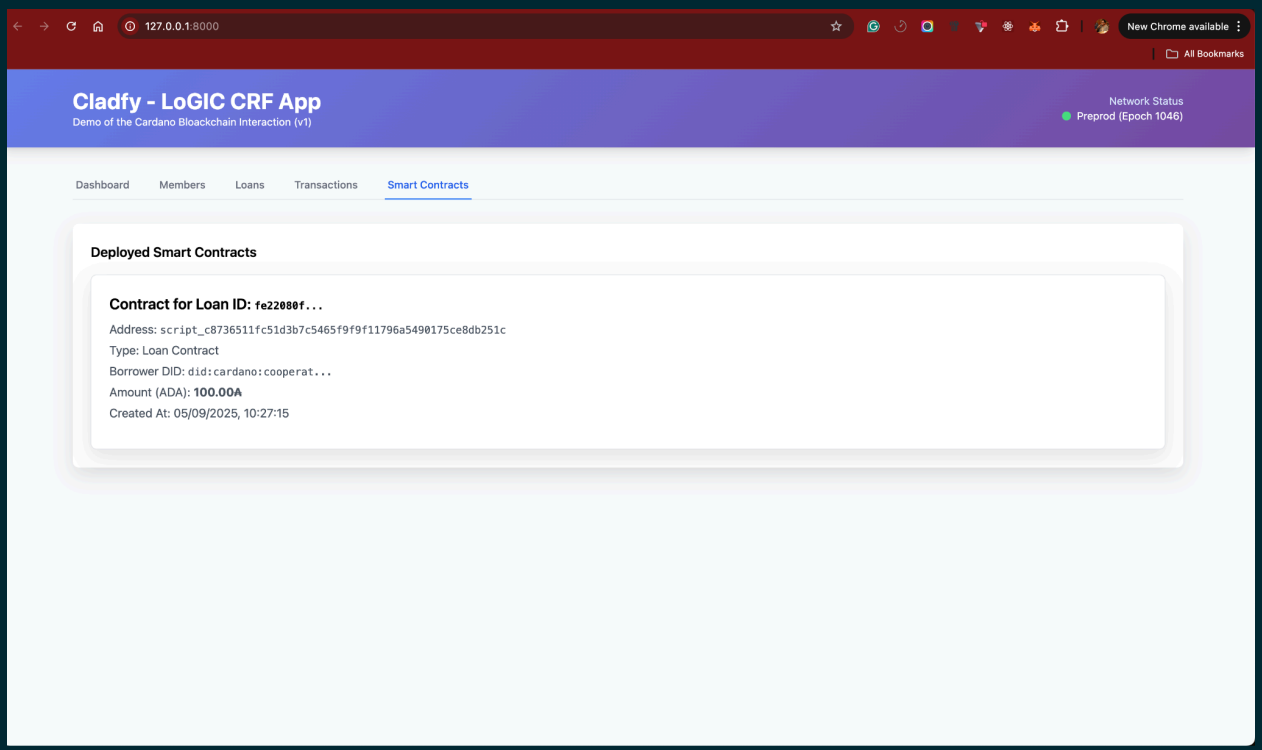
- Preprod ADA in funding wallet
- Running cardano-node synced with Preprod
- Blockfrost API key

7. Testnet / Emulator Results

Transaction ID	Contract Action	Status	Notes	Transaction ID	Contract Action
1239a7673a799df2e...	Loan Disbursement	Success	Submitted via Flask → Blockfrost integration	1239a7673a799df2e...	Loan Disbursement
c8736511fc51d3...	Loan Contract Created	Success	Generated conceptual Plutus script	c8736511fc51d3...	Loan Contract Created




preview.cardanoscan.io/transaction/1239a7673a799df2e36814ba634379688917768f26f5ce7c2dfaf39facec6735
New Chrome available
All Bookmarks
Cardanoscan
(Preview)
Home
Governance
Blockchain
Tokens
Pools
Certificates
Developers
Preview
All Filters
Search transaction, address, block, epoch.slot, pool, stakeKey, policyId.assetN
Search
Transaction Details
Transaction Hash
1239a7673a799df2e36814ba634379688917768f26f5ce7c2dfaf39facec6735
Timestamp
Sep 5, 2025 10:27:35 AM
Block
3579256
Total Fees
0.180725
Assurance
High
10 confirmations
Total Output
4,455.301797
Epoch / Slot
1046 / 26855
Certificates
0
Absolute Slot
90401255
Summary
UTXOs
Metadata (1)
Address
addr_test1vrs7yfg6slh9dct4q3595a9u3l3ldda7hhg9a3wg9pd2sqcsufnxc
ADA Spent
−0.180725


127.0.0.1:8000
New Chrome available
All Bookmarks
Cladfy - LoGIC CRF App
Demo of the Cardano Bloackchain Interaction (v1)
Network Status
Preprod (Epoch 1046)
Dashboard
Members
Loans
Transactions
Smart Contracts
Deployed Smart Contracts
Contract for Loan ID: fe22080f...
Address: script_c8736511fc51d3b7c5465f9f9f11796a5490175ce8db251c
Type: Loan Contract
Borrower DID: did:cardano:cooperat...
Amount (ADA): 100.00₳
Created At: 05/09/2025, 10:27:15

8. Dependencies & Environment

Dependencies:

```
Flask
blockfrost-python
pycardano
requests
cbor2
python-dotenv
```

Environment variables:

```
BLOCKFROST_PROJECT_ID="**********"
COOPERATIVE_TEST_ADDRESS="**********"
```

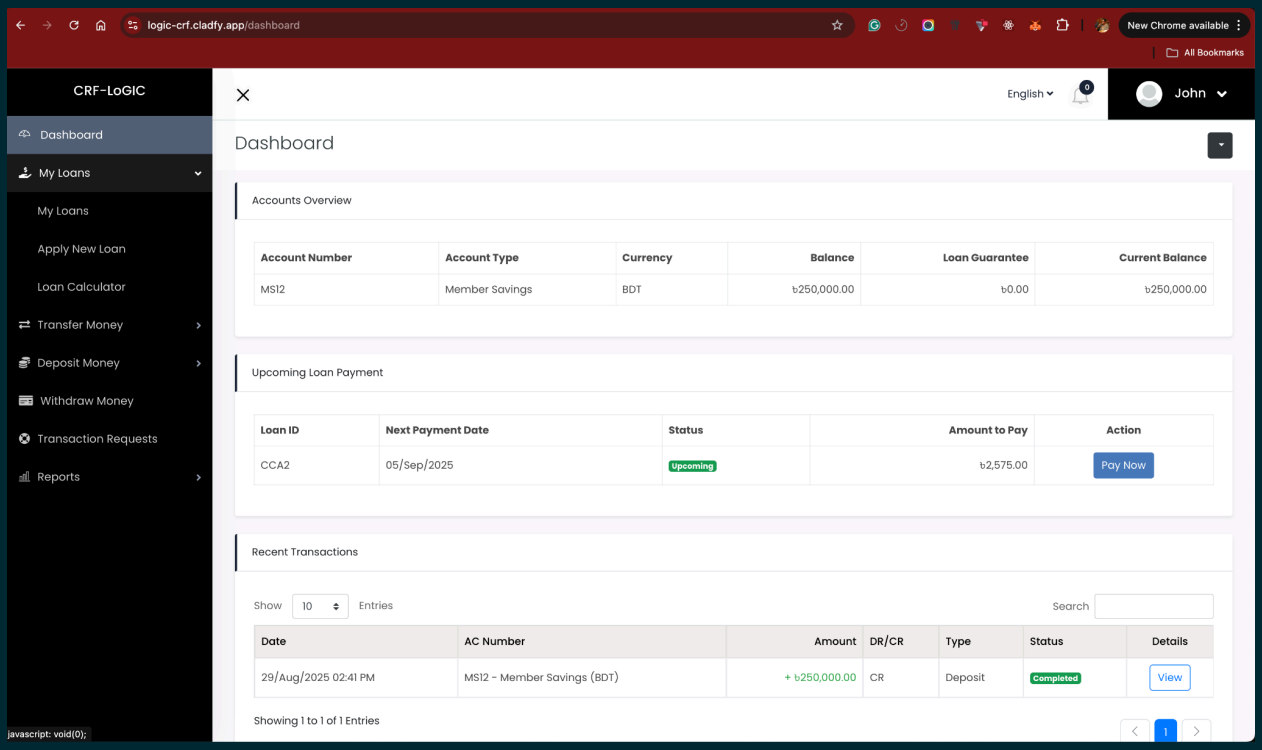
9. Demo / Walkthrough

- Register a new cooperative member → DID created (`did:cardano:cooperative:...`)
- Submit a loan application → conceptual Plutus contract generated
- Approve and disburse → Cardano Preprod transaction hash returned
- Dashboard shows loan status and repayment records

- Video Walkthrough: https://www.awesomescreenshot.com/video/43887999

- See screenshots of the Beneficiaries' access web app below.

- To access the beneficiaries' web app, visit https://logic-crf.cladfy.app



10. Remaining Issues / Next Steps

- **Persistence**: Move from in-memory storage → PostgreSQL
- **Multi-Cooperative Support**: Add registry for multiple co-ops and committees
- **Smart Contracts**: Improve contract validators
- **Integrations**: Link with the current Multi-Cooperative management system
- **Governance**: Add DAO-style voting in Phase 2
- **Security Audit**: Implement HSM/multi-sig for cooperative key management.
- **Scaling**: Benchmark disbursement throughput and transaction confirmation delays.