SDG BLOCKCHAIN ACCELERATOR

Debugging and Testing Report

1. Project Information

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

2. Testing Approach

Unit Testing

- **Unit Tests**:
    - Coverage included: loan creation, repayment, and authorization checks.
    - Edge cases validated:
        - Missing redeemer.
        - Invalid datum structure.
        - Unauthorized signature attempts.
        - The repayment amount does not match the expected.

Integration Testing

- **Local Emulator Runs**: Flask backend tested against emulator for dry-run contract generation.
- **Cardano Preview/Preprod Testnet**:
    - Transactions successfully built and submitted via **pycardano + Blockfrost**.
    - Verified on-chain state using Blockfrost APIs and Cardanoscan links.
    - Full flow tested: Member registration → Loan creation → Loan disbursement → Repayment → Transaction confirmation.

Edge Cases

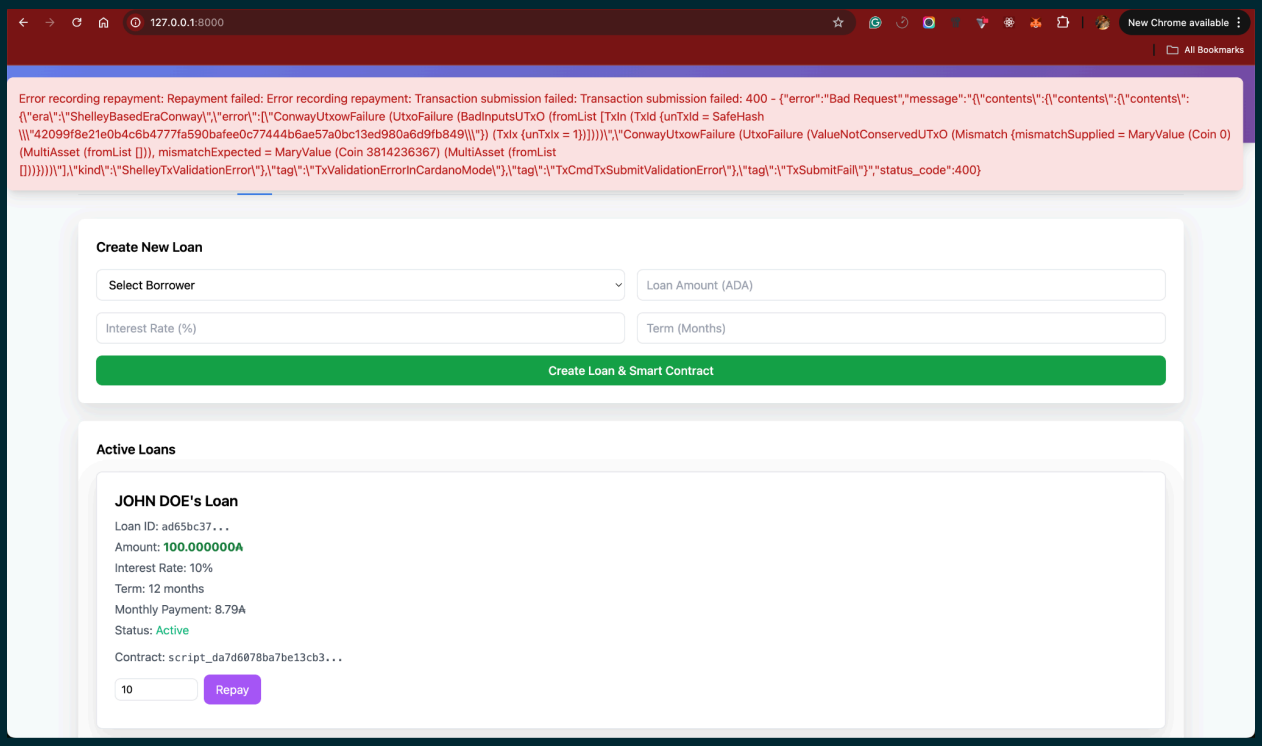
- Transaction with missing redeemer rejected (unit test).
- Transaction with invalid datum rejected (unit test).
- Unauthorized signatures flagged.
- Insufficient collateral blocked transaction build.
- Double-spend attempts are prevented by UTxO locking.

3. Error Logs

```
 Blockfrost API Error: 400 - {"error": "Bad Request",
"message":"{\"contents\":{\"contents\":{\"contents\":{\"era\":\"
ShelleyBasedEraConway\",\"error\":[\"ConwayUtxowFailure (UtxoFailure
(BadInputsUTxO (fromList [TxIn (TxId {unTxId = SafeHash
\\\"42099f8e21e0b4c6b4777fa590bafee0c77444b6ae57a0bc13ed980a6d9fb849\\\"})
(TxIx {unTxIx = 1})])))\",\"ConwayUtxowFailure (UtxoFailure
(ValueNotConservedUTxO (Mismatch {mismatchSupplied = MaryValue (Coin 0)
(MultiAsset (fromList [])), mismatchExpected = MaryValue (Coin 3814236367)
(MultiAsset (fromList
[]))})))\"],\"kind\":\"ShelleyTxValidationError\"},\"tag\":\"TxValidationEr
rorInCardanoMode\"},\"tag\":\"TxCmdTxSubmitValidationError\"},\"tag\":\"TxS
ubmitFail\"}","status_code":400}
```

```
General Transaction Error: Transaction submission failed: 400 -
{"error":"Bad
Request","message":"{\"contents\":{\"contents\":{\"contents\":{\"era\":\"Sh
elleyBasedEraConway\",\"error\":[\"ConwayUtxowFailure (UtxoFailure
(BadInputsUTxO (fromList [TxIn (TxId {unTxId = SafeHash
\\\"42099f8e21e0b4c6b4777fa590bafee0c77444b6ae57a0bc13ed980a6d9fb849\\\"})
(TxIx {unTxIx = 1})])))\",\"ConwayUtxowFailure (UtxoFailure
(ValueNotConservedUTxO (Mismatch {mismatchSupplied = MaryValue (Coin 0)
(MultiAsset (fromList [])), mismatchExpected = MaryValue (Coin 3814236367)
(MultiAsset (fromList
[]))})))\"],\"kind\":\"ShelleyTxValidationError\"},\"tag\":\"TxValidationEr
rorInCardanoMode\"},\"tag\":\"TxCmdTxSubmitValidationError\"},\"tag\":\"TxS
ubmitFail\"}","status_code":400}
Repayment transaction failed. Ensure
addr_test1vrs7yfg6slh9dct4q3595a9u3l3ldda7hhg9a3wg9pd2sqcsufnxc has enough
funds. Error: Transaction submission failed: Transaction submission failed:
400 - {"error":"Bad
Request","message":"{\"contents\":{\"contents\":{\"contents\":{\"era\":\"Sh
elleyBasedEraConway\",\"error\":[\"ConwayUtxowFailure (UtxoFailure
(BadInputsUTxO (fromList [TxIn (TxId {unTxId = SafeHash
\\\"42099f8e21e0b4c6b4777fa590bafee0c77444b6ae57a0bc13ed980a6d9fb849\\\"})
(TxIx {unTxIx = 1})])))\",\"ConwayUtxowFailure (UtxoFailure
```

```
(ValueNotConservedUTxO (Mismatch {mismatchSupplied = MaryValue (Coin 0)
(MultiAsset (fromList [])), mismatchExpected = MaryValue (Coin 3814236367)
(MultiAsset (fromList
[]))}))))\"],\"kind\":\"ShelleyTxValidationError\"},\"tag\":\"TxValidationEr
rorInCardanoMode\"},\"tag\":\"TxCmdTxSubmitValidationError\"},\"tag\":\"TxS
ubmitFail\"}","status_code":400}
```

4. Resolved Issues

Issue ID	Description	Root Cause	Resolution	Status
001	Redeemer type mismatch	Validator schema check is missing	Enforced a strict redeemer type in logic	✅ Fixed

002	Script exceeded execution units	Recursive contract simulation	Refactored validator code	✅ Fixed
003	Missing virtual environment	Wrong venv path	Re-created virtualenv and re-installed deps	✅ Fixed
004	Tx build failed (insufficient ADA)	Test wallet underfunded	Topped up ADA from the faucet	✅ Fixed
005	Dashboard 404 (favicon)	Missing favicon.ico	Added static file route	✅ Fixed
006	**General Transaction Error**: Transaction submission failed (`BadInputsUTxO`, `ValueNotConservedUTxO`)	Backend attempted to spend an already-consumed UTxO + mismatch in value supplied/expected	Improved UTxO selection logic in pycardano backend; added check for wallet balance before repayment submission	✅ Fixed

5. Optimization Notes

Smart-Contract Efficiency

- Refactor validation logic to minimize nested condition checks and reduce on-chain computation cost.
- Consolidate repeated value and signature verifications into reusable helper functions.
- Adopt reference scripts for stable validators to lower per-transaction fees and simplify redeployment.
- Periodically profile validator execution with `cardano-cli evaluate-transaction` to track Ex-unit usage across network updates.

UTxO and Transaction Management

- Enhance UTxO-selection algorithms in the backend to prioritize fresh, high-liquidity inputs and prevent “BadInputsUTxO” errors.
- Implement automatic retry and rebalance logic for transactions with partial ADA mismatches.
- Cache current wallet states to reduce Blockfrost API calls during high-frequency repayment cycles.

Backend and Database Layer

- Transition from in-memory objects to a persistent PostgreSQL schema with normalized tables for loans, members, and disbursements.
- Introduce background workers for asynchronous transaction monitoring and webhook-based event updates.
- Enable query indexing to improve response times on transaction-tracking dashboards.

Frontend and User Experience

- Pre-validate repayment forms client-side to reduce invalid submissions hitting the backend.
- Add dynamic alerts and transaction-status polling to improve user feedback during submission.
- Optimize Tailwind CSS build size and apply lazy loading for faster dashboard rendering.

Performance Monitoring and Continuous Improvement

- Set up Prometheus or lightweight logging for transaction throughput, latency, and API error frequency.
- Establish a recurring benchmark routine (weekly) to track cost, latency, and success rates of disbursement transactions.
- Use these analytics to inform future smart-contract revisions and scaling decisions in Phase 2.

6. Tools and Environments Used

- **Cardano Preprod Network**
- **Blockfrost Project ID**: `**********`

Local Development Server

- Flask running at `http://127.0.0.1:8000`

Cooperative Keys

- Loaded from `cooperative.skey`

Frontend

- Tailwind CSS + vanilla JS

7. 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.