SDG BLOCKCHAIN ACCELERATOR

Thallo Prototype (PoC) Report

1. Project Information

- **Project Name:** Thallo
- **Challenge & UNDP Office:** Tanzania
- **Document Version:** V1

2. Project Overview

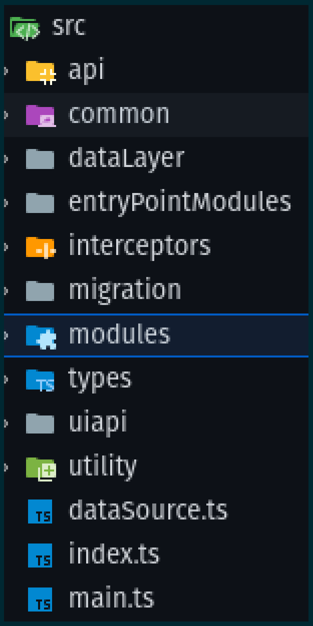
Thallo is an all in one carbon credit marketplace exchange. It allows entities to sell, buy, and retire carbon credits all on an intuitive front end. Whitelisted issuers can also issue carbon credits directly with Thallo. The credits then become available in their inventory for listing for sale.

3. Repository Structure

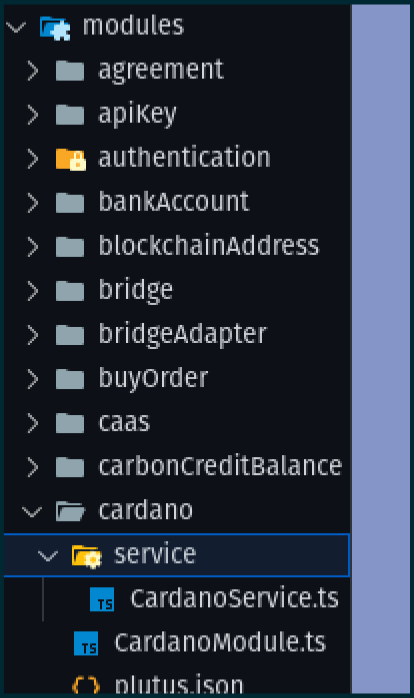
```
Carbon_credit_sc (Validators)

/lib
    /types -> Aiken smart contract types
    /utils -> Aiken test utilities
/validators
    /credits -> Mint validator to mint and retire carbon credits
    /registry -> Project registry validator for projects
    /vintage -> Vintage registry validator for vintages
/scripts -> Typescript scripts for interacting with smart contracts
```

Exchange backend is very in depth, but it uses a NestJS modules/services architecture

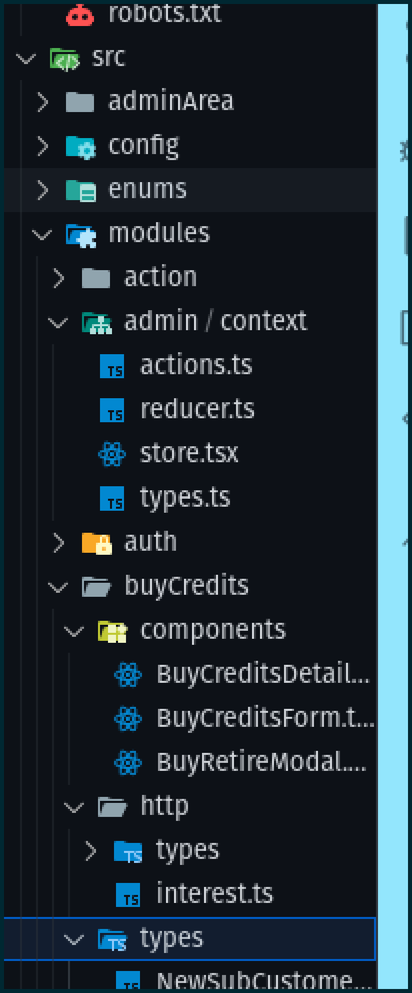


```
src
› api
› common
› dataLayer
› entryPointModules
› interceptors
› migration
› modules
› types
› uiapi
› utility
  dataSource.ts
  index.ts
  main.ts
```



```
⌄ modules
  › agreement
  › apiKey
  › authentication
  › bankAccount
  › blockchainAddress
  › bridge
  › bridgeAdapter
  › buyOrder
  › caas
  › carbonCreditBalance
  ⌄ cardano
    ⌄ service
        CardanoService.ts
      CardanoModule.ts
      plutus.json
```

Exchange frontend is a react app that tries to mimic the modules architecture to keep things organized



```
  robots.txt
src
  adminArea
  config
  enums
  modules
    action
    admin / context
      actions.ts
      reducer.ts
      store.tsx
      types.ts
    auth
    buyCredits
      components
        BuyCreditsDetail...
        BuyCreditsForm.t...
        BuyRetireModal....
      http
        types
        interest.ts
      types
        NewSubCustome...
```

4. Build Instructions

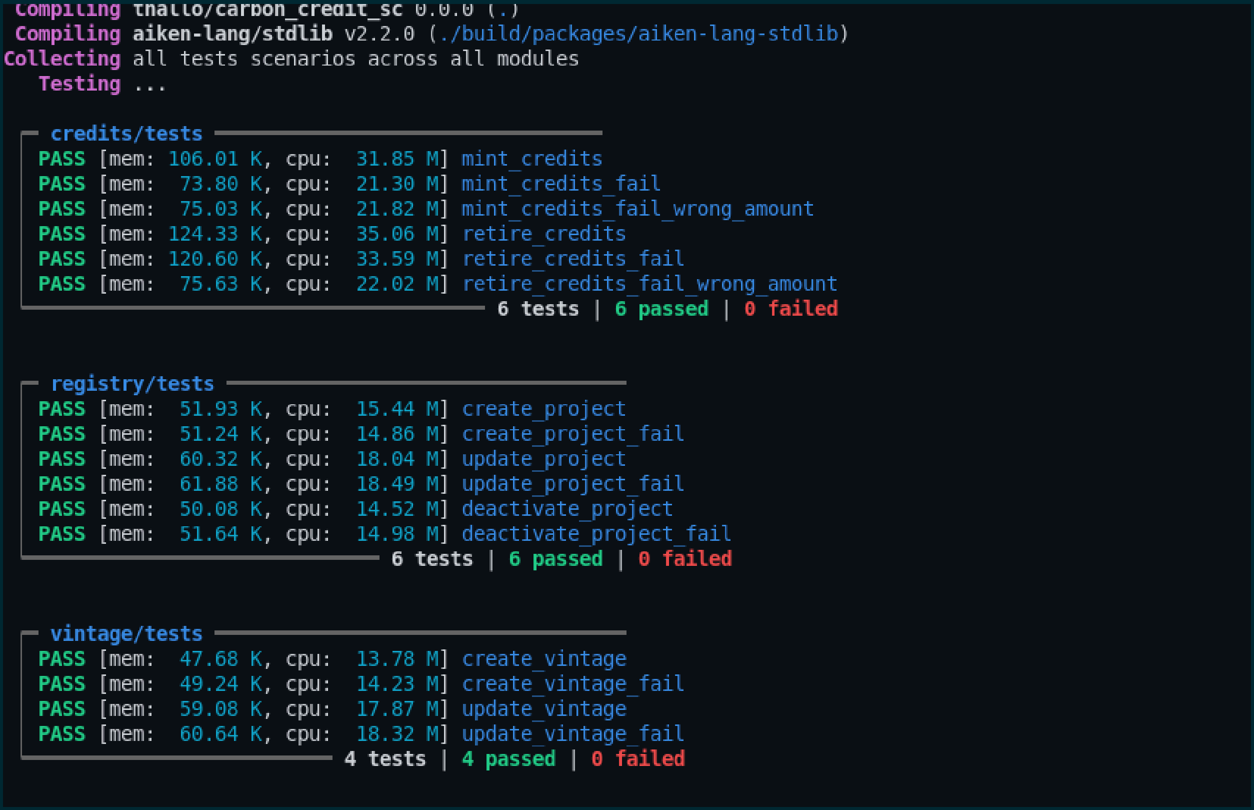
`aiken build` builds the smart contracts

`npm run start:devenv:clean:full` starts the exchange backend

`npm start` starts the frontend

5. Test Instructions

`aiken test` executes tests in the smart contract repo



```
 Compiling thallo/carbon_credit_sc 0.0.0 (.)
 Compiling aiken-lang/stdlib v2.2.0 (./build/packages/aiken-lang-stdlib)
Collecting all tests scenarios across all modules
   Testing ...

  ┍━ credits/tests ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
  │ PASS [mem: 106.01 K, cpu:  31.85 M] mint_credits
  │ PASS [mem:  73.80 K, cpu:  21.30 M] mint_credits_fail
  │ PASS [mem:  75.03 K, cpu:  21.82 M] mint_credits_fail_wrong_amount
  │ PASS [mem: 124.33 K, cpu:  35.06 M] retire_credits
  │ PASS [mem: 120.60 K, cpu:  33.59 M] retire_credits_fail
  │ PASS [mem:  75.63 K, cpu:  22.02 M] retire_credits_fail_wrong_amount
  ┕━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6 tests | 6 passed | 0 failed


  ┍━ registry/tests ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
  │ PASS [mem:  51.93 K, cpu:  15.44 M] create_project
  │ PASS [mem:  51.24 K, cpu:  14.86 M] create_project_fail
  │ PASS [mem:  60.32 K, cpu:  18.04 M] update_project
  │ PASS [mem:  61.88 K, cpu:  18.49 M] update_project_fail
  │ PASS [mem:  50.08 K, cpu:  14.52 M] deactivate_project
  │ PASS [mem:  51.64 K, cpu:  14.98 M] deactivate_project_fail
  ┕━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6 tests | 6 passed | 0 failed


  ┍━ vintage/tests ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
  │ PASS [mem:  47.68 K, cpu:  13.78 M] create_vintage
  │ PASS [mem:  49.24 K, cpu:  14.23 M] create_vintage_fail
  │ PASS [mem:  59.08 K, cpu:  17.87 M] update_vintage
  │ PASS [mem:  60.64 K, cpu:  18.32 M] update_vintage_fail
  ┕━━━━━━━━━━━━━━━━━━━━━━━━━━ 4 tests | 4 passed | 0 failed
```

6. Deployment Instructions

You will need preprod/testnet ADA and a private key to put in the environment variable for running the scripts

To execute the project registry and vintage validators, you will need to bootstrap a UTXO using the bootstrap scripts

`npx ts-node scripts/bootstrapRegistry.ts` runs the UTXO bootstrap for the registry
`npx ts-node scripts/registerProject.ts` runs the script to register the project onchain
`npx ts-node scripts/bootstrapVintage.ts` runs the UTXO bootstrap for the vintage
`npx ts-node scripts/registerVintage.ts` runs the script to register the vintage onchain
`npx ts-node scripts/mintTokens.ts` mints tokens onchain

`npx ts-node scripts/retireTokens.ts` retires tokens onchain

7. Testnet / Emulator Results

https://preprod.cardanoscan.io/address/addr_test1vpcpquku64qz3ltmhjfn8ya3p2pykfp0cnk95vfh85aet4qe9drpy

Contains the list of successful transactions on preprod

8. Dependencies & Environment

- Aiken CLI v1.1.9 (`aiken check`, `aiken build`, `aiken test`)
- Blockfrost for rpc calls and Cardano preprod
- Lucid-Evolution 0.4.29

  Lucid-Evolution utils 0.1.66

The backend is built using most NestJS (typescript) with a module/controller/service architecture.

The frontend is built via React.

9. Demo / Walkthrough

https://drive.google.com/file/d/150VY3CxbuYbjx6jjEtwEmSRMagn8tpa1/view?usp=drive_link

https://preprod.cardanoscan.io/transaction/1d383c67574f42ba15d59ad16c2fe3df928061ef6dcbcfb7d9c705ba0c9aba7e?tab=contracts

https://preprod.cardanoscan.io/transaction/f0e66024dbb5c5987b6deb6233c7c05a8ef5c5cfa614b88d2eb396e2d29b32a6

https://preprod.cardanoscan.io/transaction/b590b5f6a899ddd497b28ce93cf40fc191af0fbb36d0c968b1586c7e36b65a95

10. Remaining Issues / Next Steps

- Security audit and evaluation of validators
- Enhanced unit testing of validator contracts
- Possibly enhanced datum models for onchain data tracking