10 Academy Cohort A - Weekly Challenge: Week 5

End-to-End Web3 dApps: certificate generation, distribution, and value transfer with Algorand NFTs and smart contracts

OVERVIEW

Business Need

Web3 technology is inherently about the user controlled internet. It is being achieved by a growing stack of decentralized technologies, such as blockchains, smart contracts, oracles, crypto wallets, storage networks, and more.

In this project, the client is 10 Academy; the client would like to solve the challenge of ensuring that certificates are available to all trainees in a secure way, and (if possible) that certificate holders can benefit from smart contract actions now and in the future. At present, certificates are distributed as simple PDF files, without the ability to verify their authenticity nor can 10 Academy undertake smart actions with the trainees/their contracts.

10 Academy has partnered with Algorand to use the Algorand Blockchain as the foundational element of the NFT, and this must now be implemented. In this project you will build end-to-end Web3 dapps on the Algorand Blockchain that will help 10 Academy generate and distribute Non-Fungible Tokens (NFTs) as certificates that will represent the successful completion of a weekly challenge to trainees, and allow trainees with NFTs to interact with a smart contract to perform pre-defined actions.

DATA

- No particular data is given to this challenge. You are encouraged to explore and to visualise Algorand blockchain data.

EXPECTED OUTCOMES

Skills:

- Building REST and GraphQL APIs
- Building frontend and backend systems
- Connecting with Blockchain Nodes
- Connecting with Distributed File Systems
- Building Web3 dApps
- Extensive unit testing
- Advanced CI/CD

Knowledge:

- Web3 software development framework
- Blockchain and smart contracts

Communication:

- Blog writing

COMPETENCY MAPPING

The tasks you will carry out in this week's challenge will contribute differently to the 12 competencies 10 Academy identified as essential for job preparedness in the field of Data Engineering, and Machine Learning engineering. The mapping below shows the change (lift) one can obtain through delivering the highest performance in these tasks.

Competency	**Potential contributions from this week**
Professionalism for a global-level job	Articulating business values
Collaboration and Communicating	Reporting to stakeholders
Software Development Frameworks	Using Github for CI/CD, writing modular codes, and packaging
Python programming	Advanced use of python modules such as Pandas, Matplotlib, Numpy, Scikit-learn, Prophet and other relevant python packages

SQL programming	MySQL db create, read, and write
Data & Analytics Engineering	data filtering, data transformation, and data warehouse management
MLOps & AutoML	Pipeline design, data and model versioning,
Deep Learning and Machine Learning	NLP, topic modelling, sentiment analysis
Web & Mobile app programming	HTML, CSS ,Flask, Streamlit
Web3 & dApps	Building dApps with NFTs and smart contracts

TEAM

Tutors:

- Yabebal
- Emitnan
- Rehemet

LEADERBOARD FOR THE WEEK

There are 100 points available for the week.

20 points - community growth and peer support.

This includes supporting other learners by answering questions (Slack), asking good questions (Slack), participating (not only attending) daily standups (GMeet) and sharing links and other learning resources with other learners.

30 points - presentation and reporting.

10 points - interim submission (slides)

5 - Working principles of Blockchain, Smart Contract, and NFTs

5 - Introduction about Web3 technology stacks

5 - Clear plan to complete the project and summary of work done

15 points - final submission (report):

3 - Evidence to publication or submission of report in a blog

e.g. medium, linkedin or other similar platforms

1 - Style and quality of report (e.g. error free, font and format consistency)

1 - Creative articulation, clarity of content, and objective communication

10 - Clear sections on

1 - objectives of the project and the intended business value

1 - the technology stack used

2 - summary of what has been achieved, its implication,

and whether objectives of the project are met or not and why

15 points - Testing and Github Actions

6 points - Connection with Algorand Mainnet Node (final submission)

6 - Unit test quality and depth

5 points - Frontend

5 - Unit test quality and depth

4 points - Backend

4 - Unit test quality and depth

35 points - Web3 dApp building and CI/CD

15 points - interim submission (screenshot + Github link)

5 - Design

10 - Sandbox implementation

20 points - final submission (Github Link)

6 points - Connection with Algorand Mainnet Node (final submission)

6 - Connection with wallet

5 points - Frontend

5 - Functionality and responsiveness

4 points - Backend

4 - Code quality

BADGES

Each week, one user will be awarded one of the badges below for the best performance in the category below.

In addition to being the badge holder for that badge, each badge winner will get +20 points to the overall score.

Visualization - the quality of visualizations, understandability, skimmability, choice of visualization

Quality of code - reliability, maintainability, efficiency, commenting - in the future this will be CICD/CML

An innovative approach to analysis -using latest algorithms, adding in research paper content and other innovative approaches

Writing and presentation - clarity of written outputs, clarity of slides, overall production value

Most supportive in the community - helping others, adding links, tutoring those struggling

The goal of this approach is to support and reward expertise in different parts of the Machine learning engineering toolbox.

GROUP WORK POLICY

This is an individual assignment.

LATE SUBMISSION POLICY

Our goal is to prepare successful learners for a global level job. At work, deadlines are sometimes very strict - either you do it before the deadline or the company loses a substantial opportunity. Moreover, the late communication behaviour (submission in 10 Academy can be considered as progress communication to team leads), blinds team leads and CEOs and is very determinantal in hindering the success of the company.

We have set our late submission as follows

- Submissions are accepted only within the 12 hrs window - 17:00 UTC - 7:00 UTC of the submission deadline
- Frequently late submissions (exceeding 6 total late submissions) will disqualify a person from the list of trainees 10 Academy recommends to partner employers.
- Badges will be rewarded for the cumulative on-time appearances (gmeet calls, on-time assignment submissions, and other places where being on-time is important)

INSTRUCTIONS

The workflow for this week's challenge is as follows

- Read instructions and understand the business needs, the requirements, and the resources required/available to complete the project
- Setup git repository
- Understand the working principles behind Blockchain, Web3 dApps, Smart Contracts, etc.
- Build frontend, backend, and API connection wrappers.

Task 1 - Sandbox & API connections

Here are some tasks you should do as a minimum requirement

1. Read references and review literature to build a working understanding of the following key concepts
    1. What is the Web3 technology stack
        1. Understanding Web 3 — A User Controlled Internet - Coinbase
        2. Making Sense of Web 3. 👉Interested in helping build the… | by Josh Stark | L4 blog | Medium
    2. How Blockchain works
        1. Blockchain explained (reuters.com) - graphical and easy to follow

    2. Blockchain Demo (andersbrownworth.com) - build simple blockchain in browser
    3. An Introduction to Blockchain - inDepthDev
3. What a digital wallet is, how it is useful and how companies are beginning to exploit it
4. Accounts and Assets in Algorand Blockchain.

2. Setup GitHub repo
3. Choose frontend (react, pyscript, etc.) and backend (e.g. flask, fastapi, django, node.js, etc.) framework
4. Run Algorand sandbox
5. Explore Algorand dev features and endpoints using terminal, python script/notebook and Dappflow
6. Connect with testnet and query account information

Task 2 - Frontend & Backend Development

To help you speed up your development, you may start building the frontend first with dummy (static) data, and once you ensure the logic, wallet connection, and routes to different pages are working, you can connect it to the backend service.

1. Design and implement backend which consists of the following
    1. The backend logic - API endpoints using FastApi or Flask and Algorand Python SDK.
    2. The python backend code Tests (PyTest)
    3. Smart Contract tests using PyTeal (optional)
2. Design frontend that will satisfy the following user stories (roles):
    1. A staff role to issue NFTs and distribute
    2. A trainee role to be able to opt-in to the NFT certificate using his/her Public Key
    3. A staff role to receive trainee's opt-in request, and approve/decline the transfer of an NFT certificate to the trainee's Public Key
    4. A trainee to be able to check the approval or denial of their request.
    5. (optional/bonus - if smart contracts are implemented) A public role for anyone to donate allowed assets to a particular NFT owner.
    6. (optional/bonus - if smart contracts are implemented) A trainee who is owner of an NFT certificate to request an action (e.g withdraw token if available) allowed by the smart contract.

3. Connect backend to frontend - data in the frontend comes by calling API endpoints exposed in the backend.
4. Testing and CI: GitHub actions including Algorand Sandbox (docker-compose required) to run tests on push or PR
5. Integrate a wallet based authorisation in the frontend: Wallet Integration

Task 3 - NFT creation & Smart Contract

1. Back-End: connection to Algorand Node (local host: Sandbox, public TestNet: PureStake)
2. Back-End: connection to an IPFS Node (Web3Storage or Pianta)
3. NFT creation: follow the ARC3 (application demo, GitHub open repo) Standard
4. (optional/bonus) Implement an Algorand smart contract that is bound to the NFT assets and which will accept and distribute donations to trainees who owns the corresponding NFTs (Based on this explanation and this example - github code)

Task 4 - Blog Reporting

Write a blog-like report that details the process followed, challenges faced, product produced, and lessons learnt from this week's challenge.

N.B for reporting

Your report should start with the Introduction, the overall body of your report, and conclusion.

TUTORIALS SCHEDULE

In the following, the colour **Bold** indicates morning sessions, and *italic* indicates afternoon sessions.

Tuesday: Blockchain & Web3

Here the students will understand the week's challenge and the concepts of blockchain and Web3.

- **Challenge going through QA (YF)**
- *Blockchain Concepts & Web3 technology stack (YF)*

Key Performance Indicators:

- Understanding different blockchain accounts

- Understanding blocks, transactions, compute cycles, and memory
- Understanding tokens and fees
- Understanding NFTs and Smart contracts
- Web3 technology stack layers: Level 1, Level 2, etc.

Wednesday: DApps & NFT

Here trainees are expected to get familiar with the Algorand blockchain and its programming concepts

- **Building a DApp with the algorand blockchain (Rehmet)**
- *Algorand NFTs and Standard Assets (Emitnan)*

Key Performance Indicators:

- Programming Algorand Decentralised Apps (DApps)
- Building Algorand NFTs

Thursday: Wallets & Web3 unit testing

Here students will understand wallets and why we need them in Web3.

- **Wallet integration: why and how? (Natnael)**
- *Advanced python testing: UnitPyTest and PyTeal (Rehmet)*

Key Performance Indicators:

- Understand what are wallets and how we generate private and public keys for algorand
- Why we need wallets
- How to allow people to use their wallets to connect to a Web3 dApp
- Test design and pytest/pyteal key features

Friday: Components of a Webapp

Here students will understand the components of a web app and learn how to build a web app..

- **How to build a WebApp with ant design, react frontend, and strapi backend (Tenx Team)**
- *Algorand smart contracts (Nardos)*

SUBMISSION

Interim: Due Wednesday 10.01 8pm UTC

1. A PDF report (max 3 pages including diagrams) aimed at an audience of those with a good general knowledge of Web2, but who are interested in learning more about Web3. Your report should include the following. Consider that this document can be an extremely useful reference to share with employers when undertaking the interview process.
    1. An overview of Web3 and its key differences with respect to Web2
    2. An explanation of the Web3 technology stack, and highlight essential layers within the stack
    3. A brief description of the components (frontend, backend, wallet, NFT, Algorand Blockchain, etc.) required for the current project. Consider, if relevant, including code snippets, UI/UX, and other designs and diagrams you have made to help readers understand.
    4. Screenshot of the transaction, account, and asset pages from Dappflow that illustrates your exploration of the Algorand sandbox locally.
    5. 3 essential links for further reading, in case a reader is interested
2. Github link submission that demonstrates
    1. Well written Readme
    2. Work in progress for integrating PyTest based unit tests
    3. Work in progress for the frontend & backend development

Final: Due Saturday 13.01 8pm UTC

1. Pdf document (to be published) or (extra/bonus) a published Blog link
    1. The process you followed to build a Web3 dApp. This should include
        1. The objective of the project
        2. A bit about the Algorand blockchain (including basic stats like its market capitalisation, number of accounts etc.)
        3. What technology you choose to build your frontend and backend.
        4. What did you implement and how
        5. What lessons you learned
        6. Future plans
2. Github link submission that demonstrates
    1. Well written Readme with guide how to install and run it
    2. Well structured codebase with tests and github actions

3. Screenshots
    1. Screenshots of your frontend features
    2. Link to your deployed dApp
    3. Screenshot to actual usage of dApp connected with Algorand Mainnet (if available)
    4. Screenshots of an NFT created.

FEEDBACK

You will receive comments/feedback in addition to a grade.

REFERENCE

Papers and Blogs

- What is a dApp (Algorand): https://developer.algorand.org/docs/get-started/dapps/
- What wallets are relevant: https://www.notboring.co/p/tokengated-commerce?s=r
- Week 6 folder reading materials

General Web3 background material

- What is a wallet? https://learn.rainbow.me/what-is-a-cryptoweb3-wallet-actually
- Web1 vs. web2 vs web3 - https://learn.rainbow.me/understanding-web3
- Another view of the Web3 stack: https://blog.coinbase.com/a-simple-guide-to-the-web3-stack-785240e557f0
- Understanding the web3 stack: https://edgeandnode.com/blog/defining-the-web3-stack
- Web3 vs Web2 - https://twitter.com/vgr/status/1457449043164991488?s=20
- Web3 vs. Web2 - implementation modalities - https://moxie.org/2022/01/07/web3-first-impressions.html
- Trust: https://a16z.com/2019/11/12/the-end-of-centralization-and-the-future-of-trust/
- Crypto Cannon: https://a16z.com/2018/02/10/crypto-readings-resources/
- Bitcoin foundational paper: https://bitcoin.org/bitcoin.pdf
- NFT Cannon: https://future.a16z.com/nft-canon/

Frontend

- bigeyex/python-adminui: Write professional web interfaces with Python. (github.com)

Backend

- FastAPI (tiangolo.com)

- Welcome to Flask — Flask Documentation (2.1.x) (palletsprojects.com)

API & Wallet Integration

- peterkrull/pythonAlgorandWallet: Algorand wallet written in Python, to allow for signing of transactions and interacting with the Algorand blockchain. Includes AlgoExplorer API integration for gathering of data and sending transactions. (github.com)
- Using Algorand's Python SDK with PureStake's API | PureStake

Git Packages

- barnjamin/arc 3.xyz (github.com) - simple but powerful example of a frontend with wallet integration to generate an NFT
- algorand/auction-demo (github.com) - demo using an on-chain NFT auction and smart contracts on the Algorand blockchain.

Algorand Examples

- Building and Deploying a Decentralised Voting System with PyTeal and React | Algorand Developer Portal
- MiniBond Issuance with PyTeal and Python SDK | Algorand Developer Portal
- Build an Algorand Web Wallet Interface Using Reach and React | Algorand Developer Portal
- Linking Algorand Stateful and Stateless Smart Contracts | Algorand Developer Portal
- AlgoRealm, a NFT Royalty Game | Algorand Developer Portal
- bara96/algo-carsharing-python: Smart Contract developed in PyTEAL, CLI application written in Python to build a dApp for carpooling with Algorand Blockchain (github.com)
- Assets and Custom Transfer Logic | Algorand Developer Portal
- Algorand parameter tables - Algorand Developer Portal
- How to Develop a dApp on Algorand? (leewayhertz.com)
- Build with Python - Algorand Developer Portal - auction example above explained