Asset Tokenization - High Level Design

Prepared by: Lalitha V S

Yellow background represents the changes made to this document.

OVERVIEW

Asset Tokenization

In general terms, the tokenization of assets refers to the process of issuing a blockchain token that digitally represents a real traceable asset. These blockchain tokens can then be traded on a secondary market.

These digital tokens can include unique hash values which could represent

- Share in a company
- Ownership of a piece of real estate
- Precious metals, stones etc.
- Financial instruments, equity, bonds, fund units, etc.
- Gift cards, vouchers etc.

There are two types of assets:

1. Fungible assets
2. Non-fungible assets

Project Scope

This project primarily deals with fungible assets. There are a wide variety of fungible assets like cryptocurrencies, land registry, company shares, gift cards, meal vouchers etc.

This project is designed for a specific use case which is **Employee Rewards Tokens**.

That being said, this project can serve as a template for any fungible assets as *ERC20 implementation by OpenZeppelin* will be used as a base for building smart contracts. This project can adapt to other fungible assets by appropriate modification of UI and smart contracts.

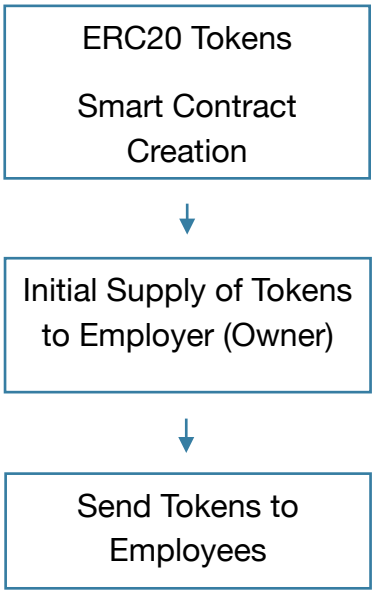
Employee Rewards Tokens

There will be two types of users:

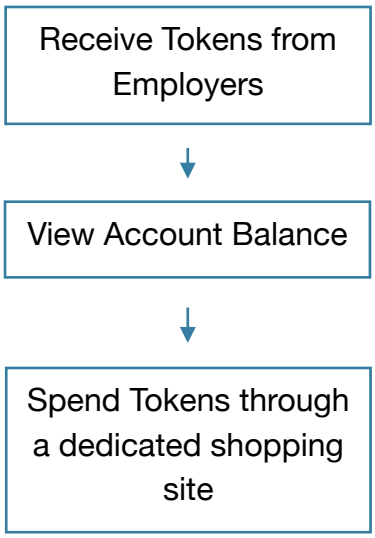
1. Employer will be
   - The owner of the initial supply of tokens.
   - Having a list of all account numbers of employees.
   - Able to send tokens to the employees, on the basis of the performance of the employees.
2. Employees will be
   - Able to view their current account balance.
   - Able to spend their tokens in exchange of some real-world objects, for example buying a meal at cafeteria.

WORK FLOW DIAGRAM

Employers

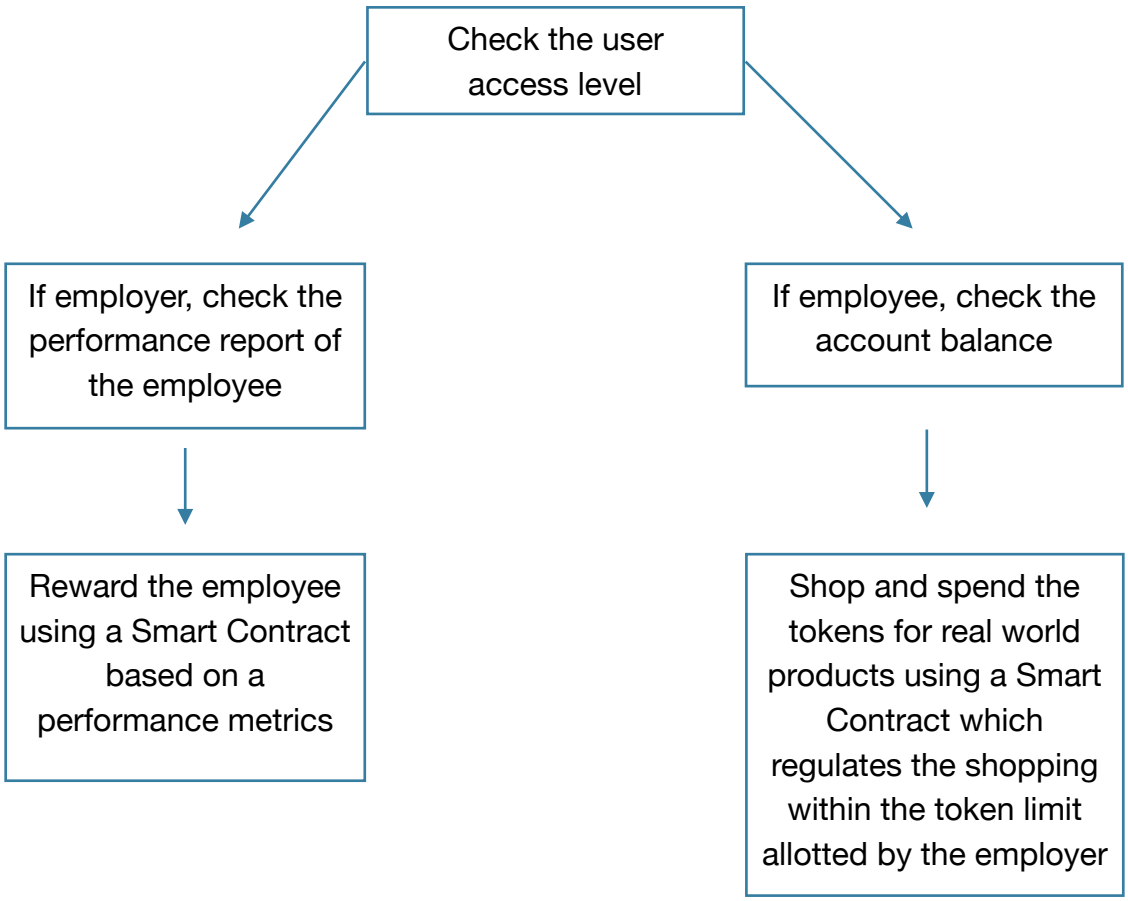


Employees



Yellow background represents the changes made to this document.

LOGIC PROCESS FLOW DIAGRAM



Note: The above diagram shows only the high level process flow as planned now. There might be changes to the flow during the development stage.

Note 2: The above high level process flow is implemented as such with the help of EmployeeRewards.sol smart contract. I had received a comment for this High Level Design document asking whether the performance metrics will be handled by the smart contract. The same has been implemented in the smart contract as well.

Note 3: Additionally, a “Disable Employee” section has been added where an employer/admin can disable an employee in case of situations like when the employee resigns.

HIGH LEVEL DESIGN

Steps

At a high level, this project is planned to be implemented in the following steps:

1. Truffle Project Initialization
2. Add in the required smart contracts from OpenZeppelin
3. Other Smart Contracts development
4. Unit Testing
5. Front end design
6. Deployment to local machine using Ganache or to a test network