# Requirements

# Mozaic System

**Development use**

**mike@mozaic.finance**

- Definition
  - Mozaic, the system, and Mozaic system refers to the software system that this project is going to develop, launche, and operate.
- Goal
  - This document serves as requirements specification of the system
  - This document serves as the technical terminology of the project.

# 1. Major concepts and requirements:

- Mozaic system is a yield aggregator. A description of yield aggregator is found at https://arxiv.org/pdf/2105.13891.pdf#:~:text=A%20yield%20aggregator%20is%20a,(see%20II%2DA).
- A user should be represented by a wallet, technically. If a user has multiple wallets, the user should be represented by the wallet that is currently involved with the system by the user's current use case.
- A user can **deposit** their assets in any listed token format on any listed chain from their wallet to the system wallet.
- The system **stake**s the deposited assets from system wallet(s) to staking pools. The system may **convert** the assets to a proper token format on a proper chain at early steps of staking. The system also **un-stake**s **stake**ed assets from staking pools. The system may **convert** the assets to a proper token format on a proper chain at later steps of un-staking.
- If the system **collect**s and **stake**s rewards generated by staking to staking pools, the system **compound**s rewards.
- *The system might not be able to track which part of an individual user's deposit is staked on which staking pool and performs how well.*
- *The system should be able convert the deposited assets to any listed token on any listed chain.*
- **Staking Stock** at a give time is the sum of the total staked assets and total pending rewards on all listed staking pools on all listed chains at the given time.
- When the system **stake**s assets that a user **deposit**ed, the system sends mLP tokens to the user in return. The amount of the returned mLP token should represent the newly staked asset in the **Staking Stock** immediately after the asset is staked.
- A user can **withdraw** assets from the **Staking Stock**, in any listed token format on any listed chain, by first sending mLP tokens from their wallet to the system wallet.
- The amount of asset that is **withdraw**n is the portion of **Staking Stock** that is represented by the returned mLP tokens immediately before the asset is **withdraw**n.

- The mLP token exists on all listed chains.
- All mLP token versions should always have the same global price.
- A user can transfer one version of mLP token from their wallet to any version of mLP token on any wallet or account.
- The system should make sure that only the system can **stake** deposited assets to staking pools, subtract **un-stake** from staking pools, collect pending rewards from staking pools, and **compound** rewards.
- The system should be able to optimize staking globally to maximize staking rewards.

## 2. Use cases

The following diagram (UML Usecase Diagram) illustrates the system requirements.

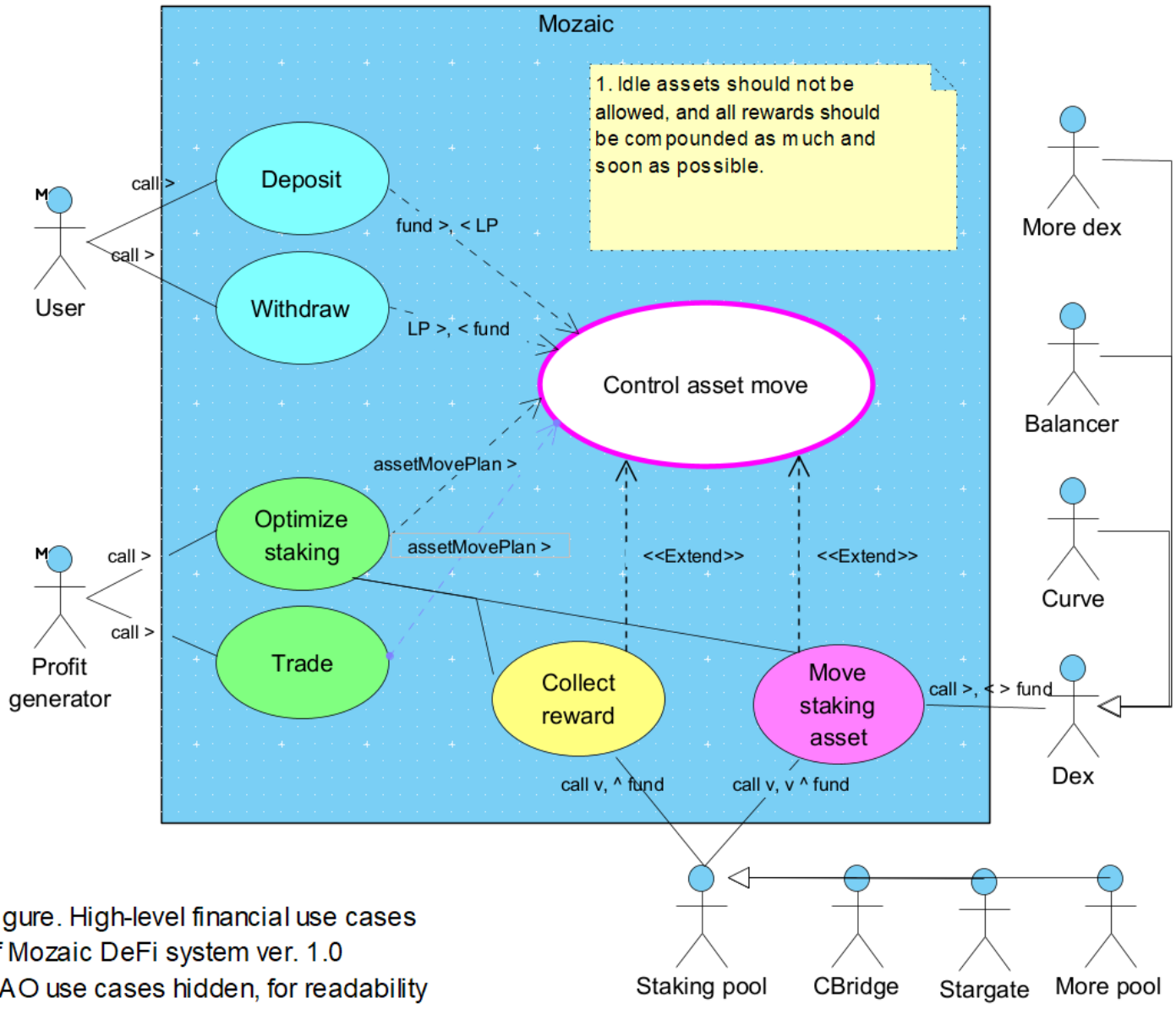


Figure. High-level financial use cases of Mozaic DeFi system ver. 1.0
DAO use cases hidden, for readability

The use cases and external actors are described as below:

- **Compound**: This hidden use case compounds rewards. All rewards should be compounded as much and soon as possible.
- **Control asset move**: This use case checks, carries out, and keeps track of asset moves. The assets managed by the system can only be moved by this use case transparently.
- **Deposit**: This use case deposits the user's assets in the system. A **User deposit**s their assets requesting that the assets will be added to **Staking Stock** and they will receive mLP tokens in return. **User** should be able to deposit any listed token on any listed chain, and receive mLP tokens on any listed chain. This use case includes **Control asset move**.
- **Withdraw**: This use case withdraws from **Staking Stock**. A **User withdraw**s part/all of their share from **Staking Stock** by first sending mLP tokens to the system, in the hope that they will earn profit yield. A **User** should be able to withdraw any listed token on any listed chain, no matter what

token on which chain they deposited, by first sending mLP tokens on any listed chain. It includes **Control asset move**.

- **Optimize staking**: This use case upgrades the staking portfolio to get better rewards. **Profit generator**, a role of the system, invokes this use case *either on a regular basis or at randomly picked times*. This use case includes **Control asset move** and **Compound**. *By providing **Control asset move** with **transitionPlan**, this use case effectively prevents it from being involved with finding optimal staking portfolio.*
- **Trade**: This use case swaps idle assets in the hopt of benefiting from price changes. It invokes **Dex**. *By providing **Control asset move** with **assetMovePlan**, this use case effectively prevents it from being involved with finding optimal trading orders.*
- **Collect reward**: This use case collects rewards from Staking pools. Use case **Control asset move**, when it is working included **Optimize staking**, is extended by this use case. This use case invokes **Staking pool**.
- **Move staking asset**: This use case moves assets to/between/from, **Staking pools**. **Control asset move**, when it is working included **Optimize staking**, is extended by this use case. This use case invokes **Staking pool**.
- **Dex**: This actor is a smart contract that swaps between assets. Examples are pairs on Curve and Balancer DeFis.
- **Staking pool**: This actor is a smart contract that allocates reward to assets that are staked in it. Examples are farming pools on CBridge and Stargate DeFis.

# 3. Lawful responsibility and freedom of the system

- **Guaranteed Withdrawal**: The system is legally obliged to allow a user to withdraw the full amount of assets that the user has deposited in the system, in XXX hours after the user's request.
- **Unguaranteed Profit**: The system is not legally obliged to return profit to a user who deposited assets in the system.
- **Guaranteed Asset Control**: No person or entity other than the system can stake deposited assets, subtract from staked assets, collect reward, and compound reward.
- **Transparent Logging**: All asset/profit creation/move must be correctly logged in/at an available format and place.
- **Transparent Profit**: The division of profit among users must be in accordance with the promise transparently.