EmpowerTheDAO - Project 1 - Scope

This document contains an overview of the scope of the applications being developed as part of the initial project by EmpowerTheDAO, an Aragon Nest project. For previous discussions regarding the proposal for this project, please see this Github issue.

Summary Scope

This project will deliver “minimum viable” integrations between Aragon and the protocols of **Uniswap**, **Compound Finance** and **Ethereum Name Service (ENS)**. These will be in the form of Aragon Agent applications, which can be installed into any DAO on the Aragon platform.

For the avoidance of doubt, and as outlined in the proposal, these applications will seek to take the first step to integrating with these protocols. The apps will not cover all aspects of the functionality of the protocols being integrated with.

The scope described in this document defines the functionality which will be released for functional testing, and for security audit.

General Scope

Each application will be able to be installed as a standalone application in the App menu for a DAO, with associated logo.

Asset management functionality to deposit to, and withdraw from the apps will be included to supply them with the assets required to perform their functions.

Permissions for which entities are allowed to perform specific functions will be delegated to the Aragon OS. Furthermore, the UI will follow Aragon’s design guidelines.

Protocol-Specific Scope

The apps will have functionality to empower an Aragon DAO to interact with these protocols:

- Compound Finance
    - Supply DAI: Deposit DAI and earn interest
    - Withdraw DAI: Withdraw DAI to stop earning interest.
- Uniswap Exchange
    - Swap ETH for DAI: converts Ether into DAI
- Ethereum Name Service
    - Set Reverse Record: sets an ENS name to resolve to the app’s address

Application Scope

This section describes the scope for the “minimum viable applications” to integrate Aragon with each of the protocols.

Compound App Scope

This section describes the scope of the app to empower any Aragon DAO to lend using Compound Finance (https://compound.finance/).

Smart Contract Functions

The following functions will be available on the application:

- Functions to manage the assets controlled by the application:
  - **Deposit DAI** - allows DAI to be deposited to the application’s address
  - **Withdraw DAI** - allows DAI to be withdrawn from the application’s address

- Functions on Compound’s protocol:
  - **Supply DAI** - allows the application to supply ETH to be lent
    - Uses `cDAI.mint()`
  - **Withdraw DAI** - allows the application to withdraw ETH from being lent
    - Uses `cDAI.redeemUnderlying()`

The framework used for supplying and withdrawing DAI can be reused during further development, for supplying and withdrawing other ERC-20 tokens which are available for supply on Compound.

Smart Contract Addresses

The smart contract addresses are defined on Compound Finance’s Developer’s portal.

Useful links:
https://github.com/compound-finance/money-market-presidio

Uniswap App Scope

This section describes the scope of the app to empower a DAO to swap assets using Uniswap's protocol (https://uniswap.io/).

Smart Contract Functions

- Functions to manage the assets controlled by the application:
    - **Deposit ETH** - allows ETH to be deposited into the application's address
    - **Withdraw ETH** - allows ETH to be withdrawn from the application's address
    - **Deposit DAI token** - allows DAI to be deposited to the application's address
    - **Withdraw DAI token** - allows DAI to be withdrawn from the application's address

- Functions on Uniswap's protocol:
    - **Swap ETH for DAI** - converts Ether into DAI
        - Uses `ethToTokenSwapOutput(uint256 tokens_bought, uint256 deadline)` sending ETH with the function
    - **Swap DAI for ETH** - converts DAI into Ether
        - Uses `tokenToEthSwapOutput(uint256 eth_bought, uint256 max_tokens_sold, uint256 deadline)` approving DAI prior to calling the function

The framework used for swapping DAI for ETH and ETH for DAIcan be reused during further development, for swapping ETH to / from other ERC-20 tokens available for swapping on Uniswap.

Smart Contract Addresses

The smart contract addresses are defined on Uniswap's integration documentation.

ENS App Scope

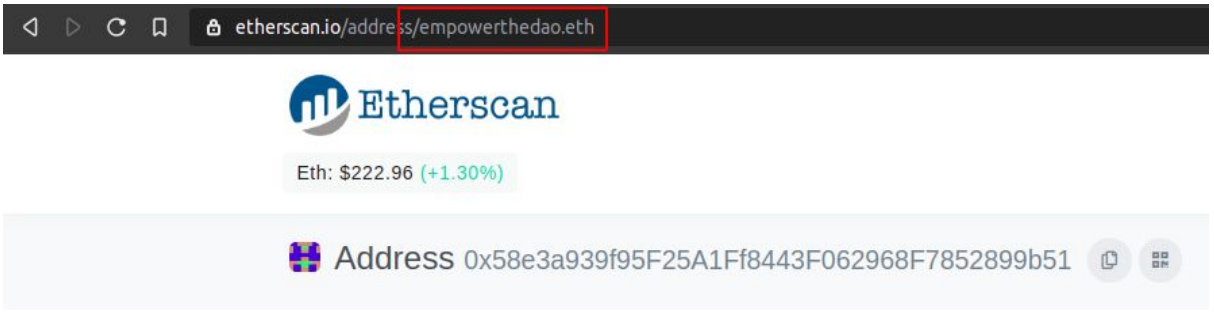
This section describes the scope of the app to empower a DAO to interact with ENS.

Smart Contract Functions - Basic Scope

- Functions to manage the assets controlled by the application:
    - *Not required, as no assets are required to be controlled by this application.*

- Functions on ENS's protocol:
    - **Set Reverse Record**: to sets an ENS name to resolve to the app's address
        - Uses `setName(string name)`
        - Here is an example transaction from Ethereum Mainnet
        - This will configure the app's address to be resolved via an ENS name

This will allow a DAO's `Agent` app address to be resolved to a human-readable ENS name:



Registering Names - Limitation of ENS Protocol

The process for registering a new name on ENS consists of two steps: `Commit()` and `Register()`. This process involves passing a "secret" from the `Commit()` function to the `Register()` function. This is designed to protect the user who is registering the name.

This process is optimised for use by an individual using an Externally Owned Account. It is therefore not possible for a DAO to register a new name on ENS, without compromising the DAO's security. This is due to the fact that discussions of the domain name would be exposed to third parties prior to it being registered, who could register it before the DAO can.

One option considered, was to assume that discussions about the domain name the DAO would register would be kept secret, and also to assume that an individual would be prepared to call `Commit()` and `Register()` through the DAO app. However it is not likely that these assumptions can be guaranteed to always be true.

Furthermore, a less complex version of this process can already be executed by an individual, to register a domain independently themselves and transfer it to the DAO app.

Smart Contract Addresses

BaseRegistrar and EthRegistrarController are found using the process described here: https://docs.ens.domains/contract-api-reference/.eth-permanent-registrar#discovery