We are working on a DAO for ETH staking.

ETH holders want to earn some interests on ETHs. There is a limitation of a minimum 32 ETH for a validator node and we want to lower this limitation by creating a set of smart contracts

Suppose there are 32 people and everyone has 1 ETH. They have to deposit the money to some address.

Smart contracts.

1. For depositing the money. When 32 ETH is reached, suppose that someone sends some ETHs I guess we should send their money back. Until 32 eth is not reached, the client should have the opportunity to get the money back.
2. When setting up the server for the PoS we will get a validator and a withdrawal key.

Validator key. We have the ownership of the key.

* 1. Multi - signature wallet. Check the trust setup. Discuss together.
  2. The custody of the withdrawal key – che ck trust setup - should be split among the 32 validators. We should create a governance contract where each of the ETH holder has a voting power – similarly to what Compound did https://compound.finance/docs/governance. When 51% of the consensus is reached, all funds will go back to the owners after one week approximately.

1. Simple website for the demo that shows the validator waiting

Documentation about the mechanics of PoS - will translate the description it in English

* A video that shows the transition PoW to PoS: phase 0, phase 1 and phase 2:

[Ethereum 2.0 is coming – Here’s what you NEED to know](https://www.youtube.com/watch?v=Z5gKKFszeAA)

* **How to launch an Ethereum validator node**

<https://docs.prylabs.network/docs/getting-started/>

           [Reference: <https://medium.com/prysmatic-labs/ethereum-2-0-phase-0-testnet-release-1e9e682db910>]

--------------------------------------------

Greater dive into PoS if necessary

* PoS FAQ:

<https://github.com/ethereum/wiki/wiki/Proof-of-Stake-FAQ>

* PoS en general

<https://docs.ethhub.io/ethereum-roadmap/ethereum-2.0/eth-2.0-economics/>

UI requirements

In UI, we need a very simple UI . it should contain the following fields .

1. List/Dropdown to display the transactions ids.
2. A button to trigger the events after selecting the transaction id from the UI.
3. List/Dropdown which will show the addresses of the users registered [to be displayed only to the contract owner].
4. A page which contains the button , which user can click to register himself.
5. A textbox which can allow user to enter the token value to be deposited and button to trigger the event
6. A textbox which can allow user to enter the amount he wants to withdraw and button to trigger the event