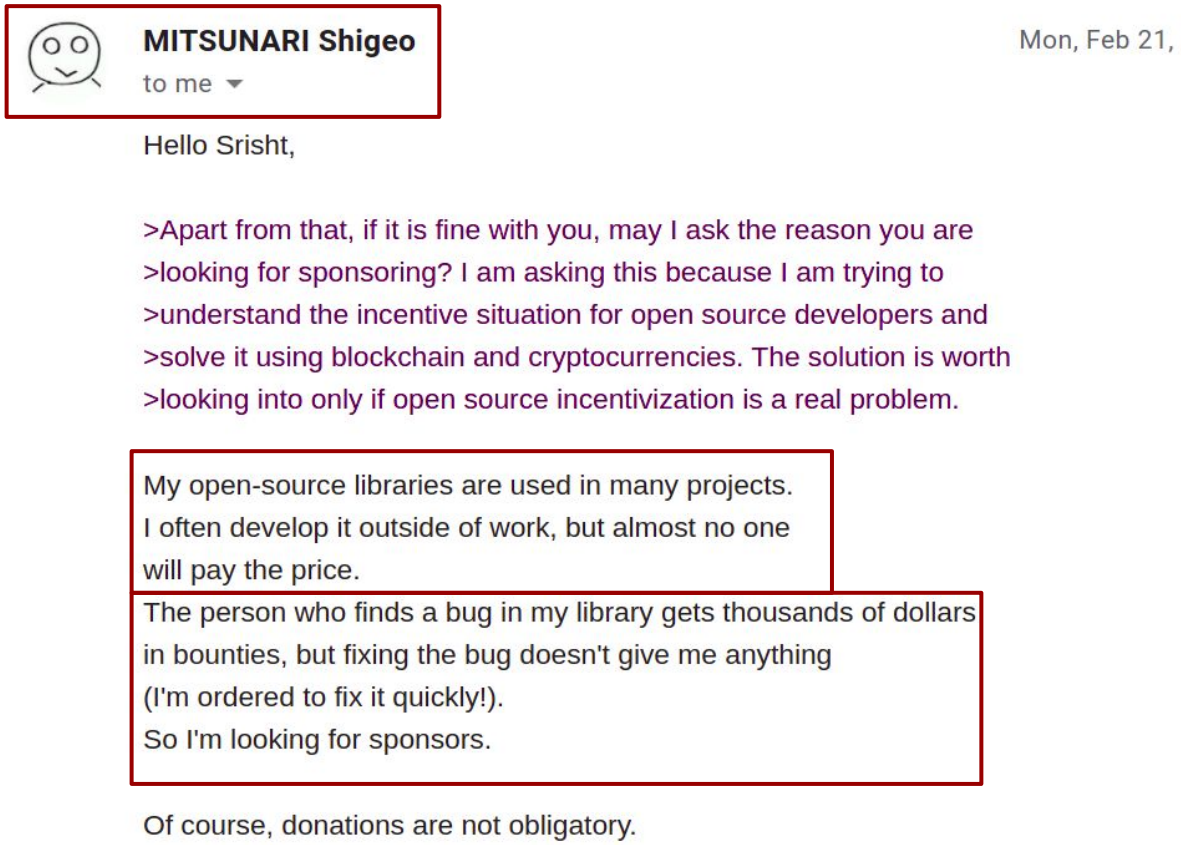
Macro-hard: A permissionless & decentralized software corporation

By: Srisht, Panos, and Hossein

Problem: No incentivisation for FOSS dev projects

- Open Source devs don't get the deserved monetary incentivisation.
- Prevalent even for blockchain projects

FOSS: Free & Open-Source Software



MITSUNARI Shigeo
to me

Mon, Feb 21,

Hello Srisht,

>Apart from that, if it is fine with you, may I ask the reason you are
>looking for sponsoring? I am asking this because I am trying to
>understand the incentive situation for open source developers and
>solve it using blockchain and cryptocurrencies. The solution is worth
>looking into only if open source incentivization is a real problem.

My open-source libraries are used in many projects.
I often develop it outside of work, but almost no one
will pay the price.
The person who finds a bug in my library gets thousands of dollars
in bounties, but fixing the bug doesn't give me anything
(I'm ordered to fix it quickly!).
So I'm looking for sponsors.

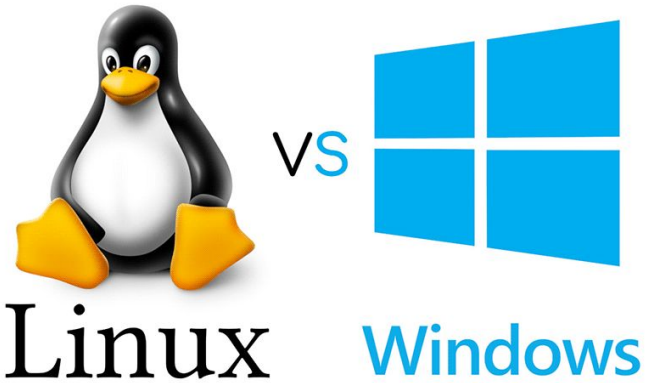
Of course, donations are not obligatory.

Problem: No incentivisation for FOSS dev projects

- Although some of them get donations on Github,
- But charity is not the right way to reward them!

Problem: No fair ownership of FOSS projects

- FOSS developers don't even get ownership for their contributions (big or small) in the projects.
- Here ownership means decision making power or intellectual property rights.



Software Corporations: Incentivisation and ownership but high barrier of entry

Corporations have

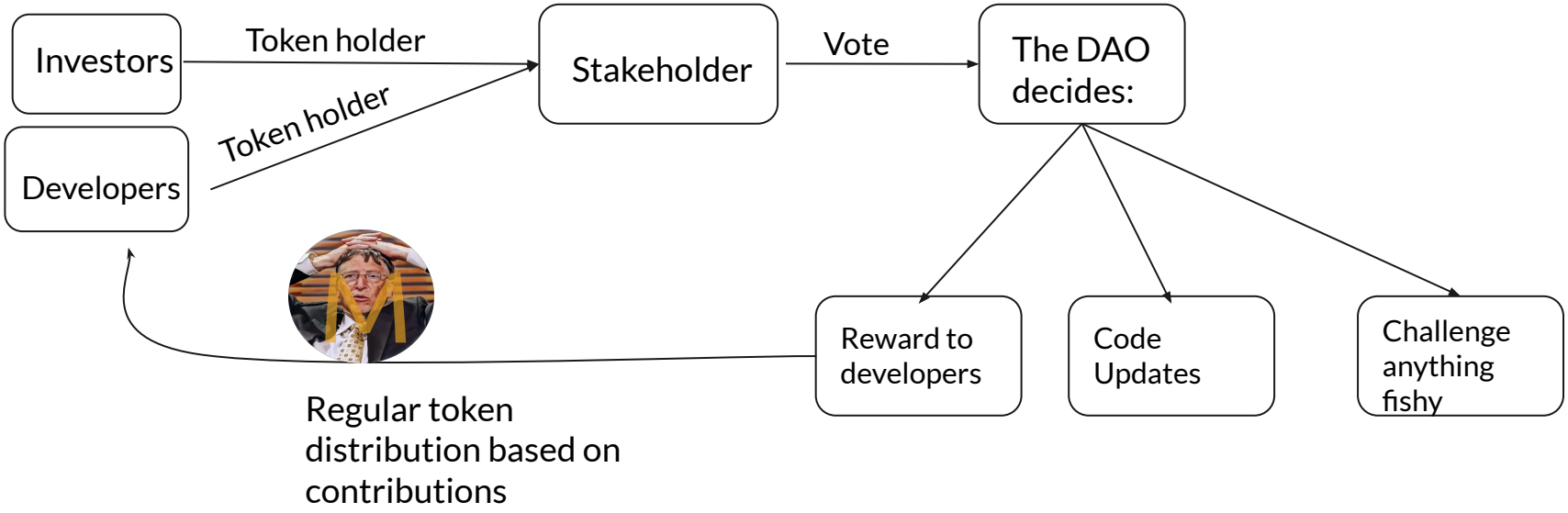
- Incentivisation,
- Sometimes IP rights, but
- Uncertain entry

Freedom for everyone to showcase their talent.

Solution: MacroHard

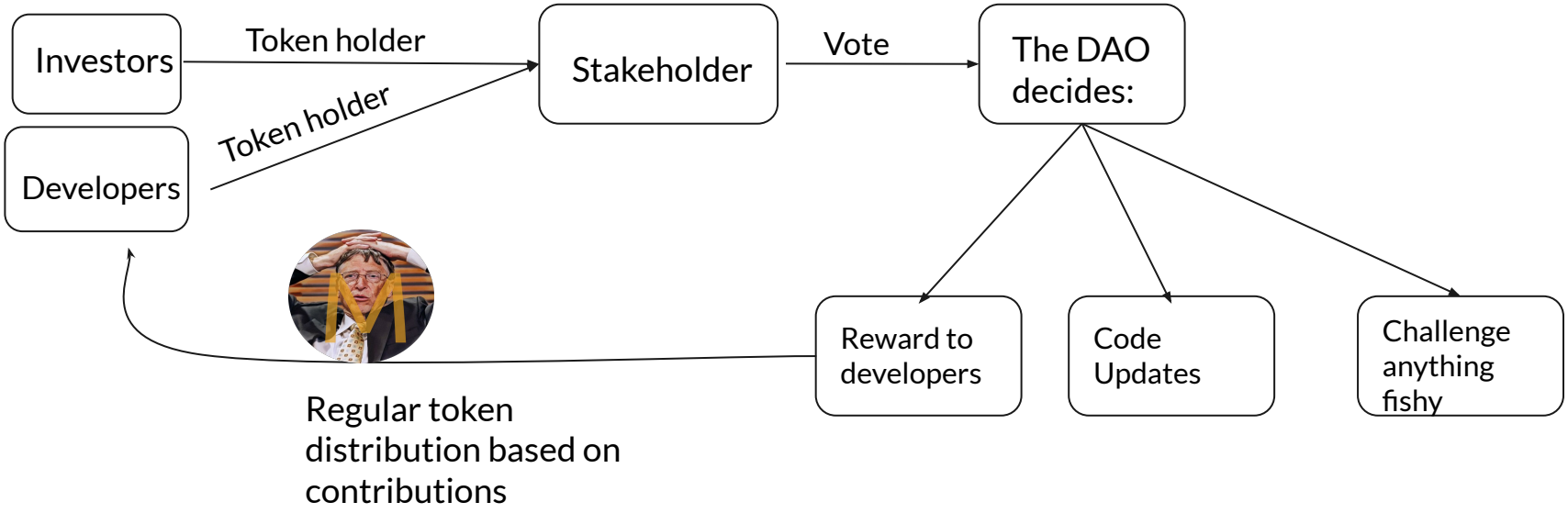
- Permissionless: anyone can start contributing to a project at any point.
- Incentivization: contributors receive native reward tokens based on their contributions.
- Ownership: project contributors can vote on technical and economical proposals in accordance to their token holdings.

Solution: MacroHard



Solution: MacroHard

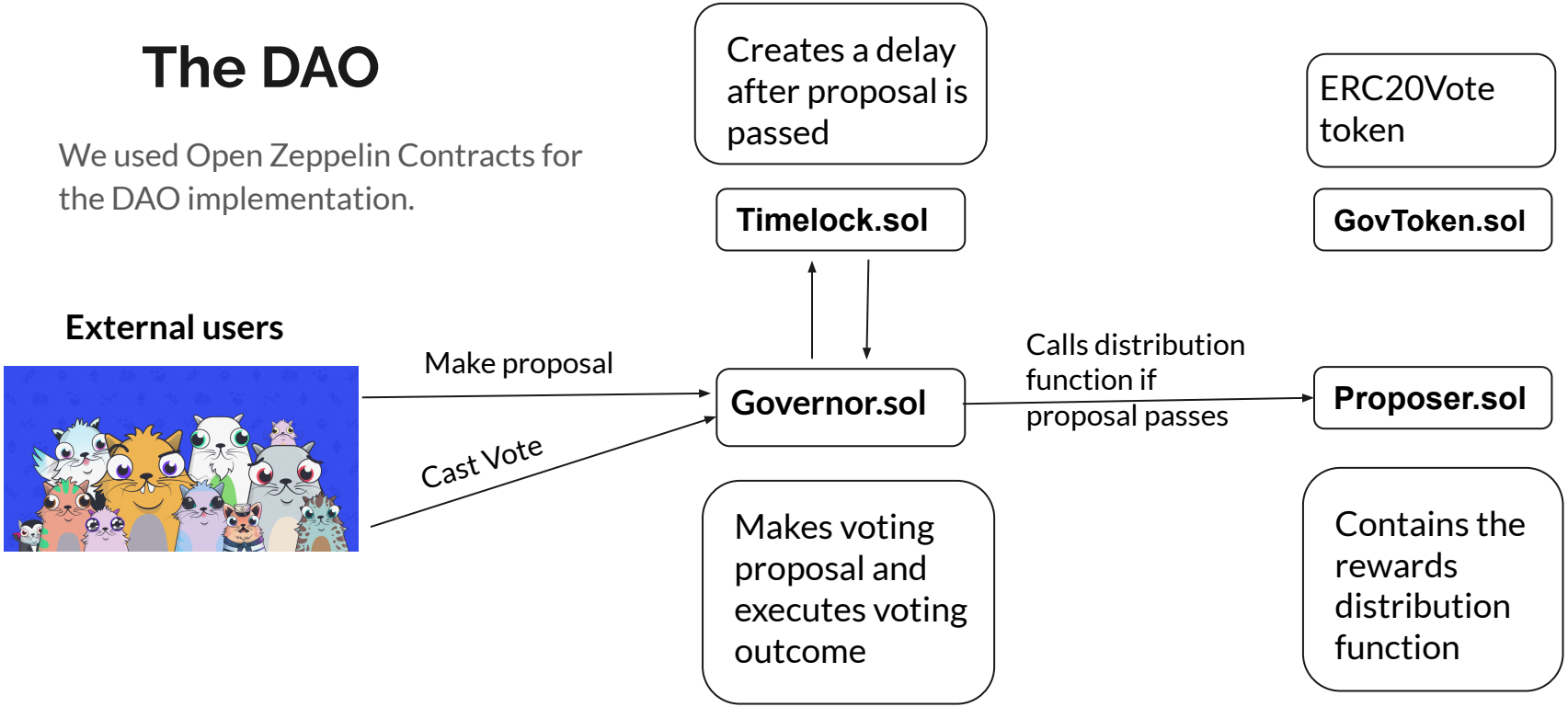
Anyone can join as a developer!!

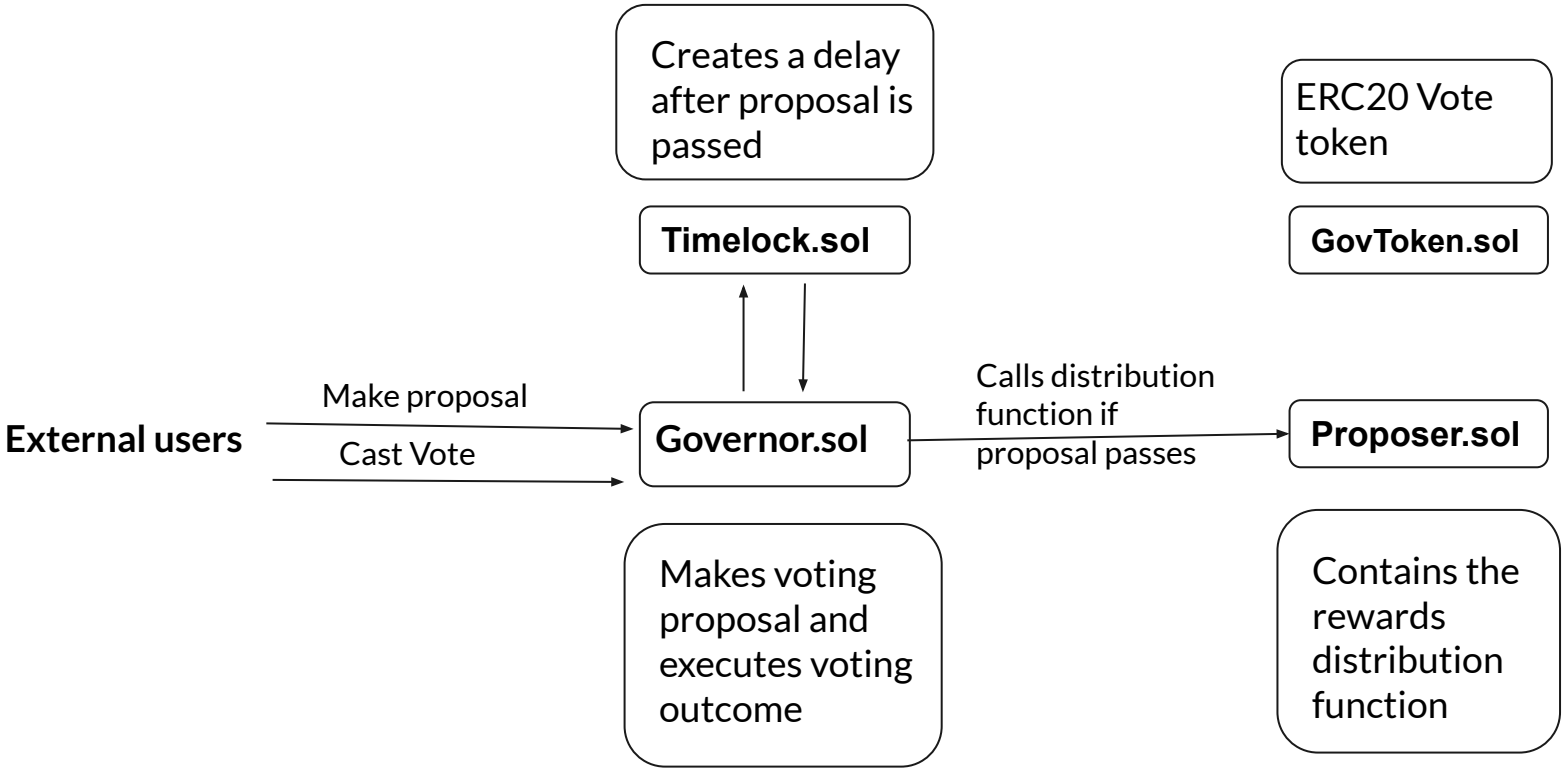


No crypto is complete without Mr. Musk

The DAO

We used Open Zeppelin Contracts for the DAO implementation.




Creates a delay after proposal is passed
ERC20 Vote token
Timelock.sol
GovToken.sol
External users
Make proposal
Cast Vote
Governor.sol
Calls distribution function if proposal passes
Proposer.sol
Makes voting proposal and executes voting outcome
Contains the rewards distribution function

ERC20Votes Token

Why not just import an ERC20 smart contract? Because an OpenZeppelin extension of it called ERC20Votes provides us with functionalities that are pretty useful when it comes to dealing with DAOs and made it a much better voting tool:

- **Snapshots** of number of tokens contributors have: We wanna avoid contributors just buying and selling tokens right before and after a hot proposal respectively to get in on decision making
- Gives useful **vote related info** (how many votes one has, their past votes, etc.)
- **Delegation**: You can delegate your tokens to others, to let them vote instead of you in case you weren't able to do so
- ...

TimeLock

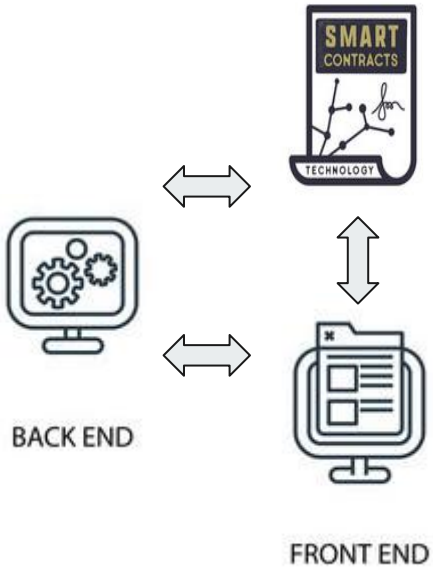
- TimeLock smart contract is the owner of the proposer contract.

  Which means it’s up to this contract to call any function in the proposer contract or not.

- Whenever we make a proposal we want to wait for a while to make sure after the new vote’s execution contributors have enough time to get out if they don’t like an update.

Implementation Design

- All the votings are implemented in smart contracts to be on the chain (e.g. Vote on the confirmation of evaluation of contributor's work)
- The calculation of contributor's work is implemented in back-end based on fetched data by Github APIs (Yeah, it's partially decentralized!)
- Front-end gets proposal information from the back-end and interacts directly with the blockchain for voting and balance retrieval



Evaluation of the amount of work

We evaluate each contributor's work based on the number of their contributions to the project's Github repository. Which obviously is against our desire of having a fully decentralized system, yet if we wanted to make it completely decentralized we should have created an on-chain source-code management and version control service as well. Because of all the limitations we decided to outsource this part and get use of a third party.

Handle attacks on centralized parts

In case an adversary wants to make an attack on the back-end or on the third party's services, the voters could easily reject the proposal (e.g. contribution evaluation) till everything goes back to normal.

You should be polite

In order to get the contributions' data from Github servers we need to send requests to Github through provided APIs. However, in case of making too many api calls, the api provider would not be able to respond them or depending on their policies they could ban your IP. To avoid that, we should simply be "polite", which means to keep the rate of api calls low.

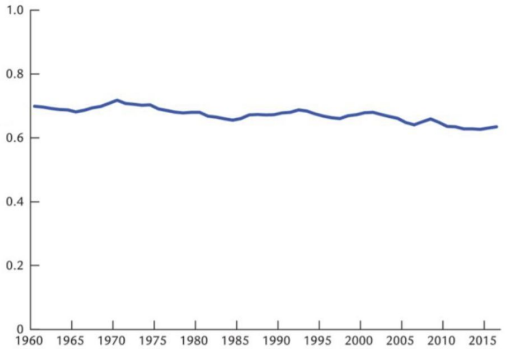
To do so we needed to use LRU cache to keep fetched data for repositories from past hour and avoid sending requests for them.

Incentives to avoid inconsistent progress

In order to keep projects' development going on in a consistent manner we designed a feature that gives the voters the ability of defining dynamic sized time windows to evaluate contributors' work within them. By normalizing the amount of work they did in each time window, we make sure to keep the work going on consistently.

If nobody does any contribution within a time window, the tokens assigned to that time window would get back to investors' pocket.

If only a few of them work in a time window they'd gain a lot of money for less work. Hence, it's not going to stay like that forever. Finally the system would reach to a state that there's a balance between supply and demand of work.



LA quick look at the initial version's functionalities

Register

PROPOSALS

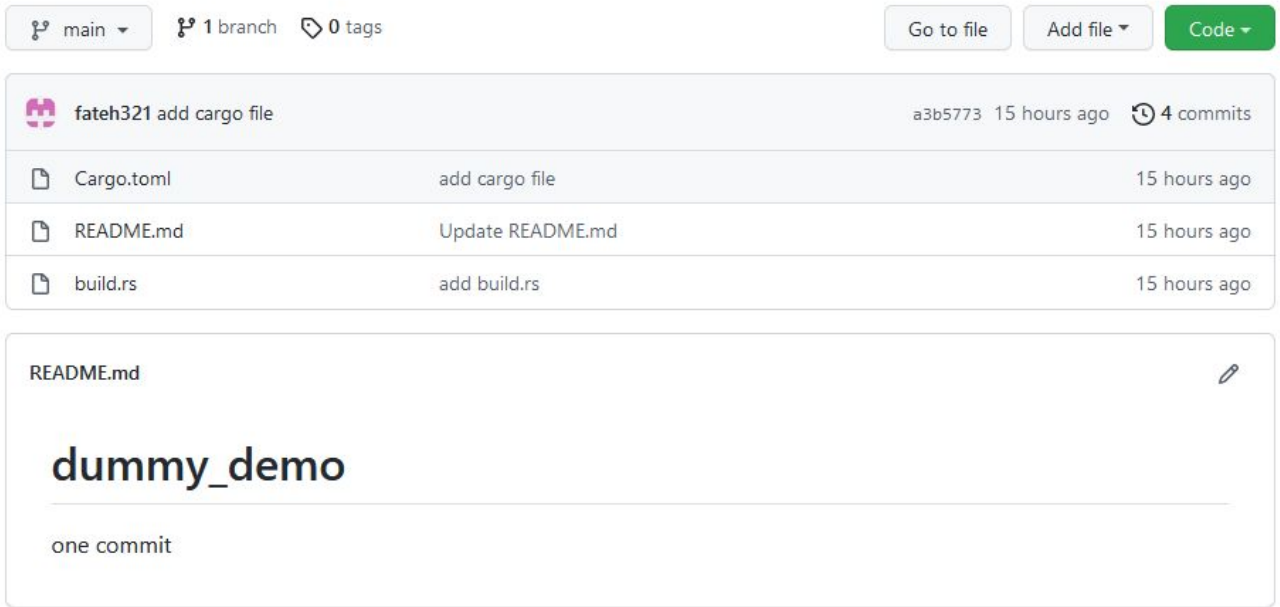
REGISTER

Github ID:

Blockchain Address:

Submit

Contribute


main
1 branch
0 tags
Go to file
Add file
Code
fateh321 add cargo file
a3b5773 15 hours ago
4 commits
Cargo.toml
add cargo file
15 hours ago
README.md
Update README.md
15 hours ago
build.rs
add build.rs
15 hours ago
README.md
dummy_demo
one commit

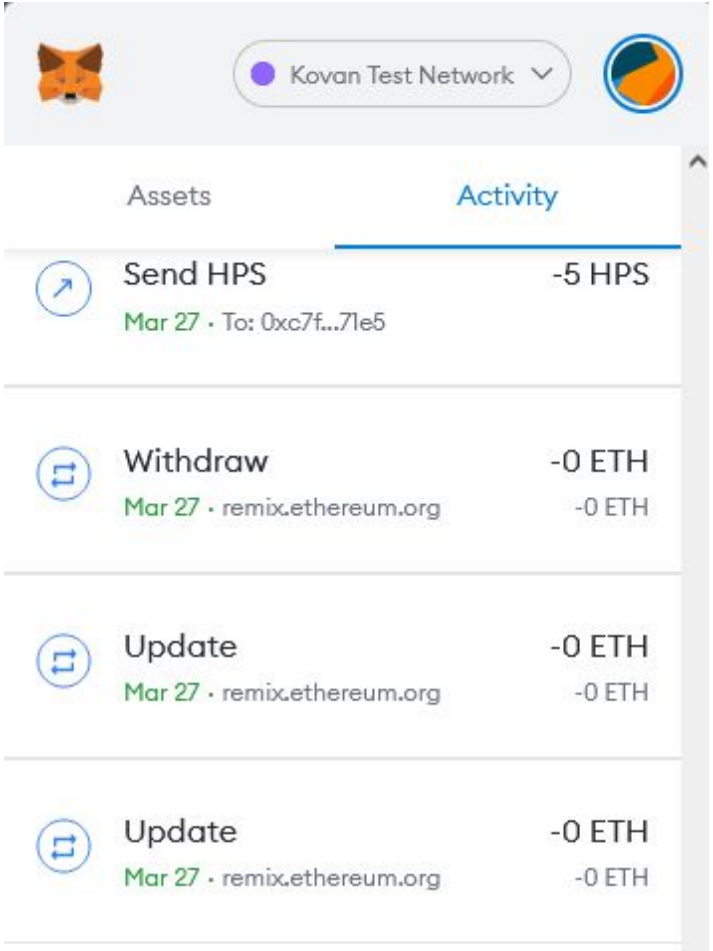
Vote

PROPOSALS REGISTER

test 10
Creation Block: 30686236
Valid for 20 blocks
End Block: 30686256
Votes for: 250000000000000000000000
Votes against: 0
Abstain: 0
Proposal status: Executed

VOTE FOR VOTE AGAINST ABSTAIN

Withdraw


Kovan Test Network
Assets
Activity
Send HPS
-5 HPS
Mar 27 · To: 0xc7f...71e5
Withdraw
-0 ETH
Mar 27 · remix.ethereum.org
-0 ETH
Update
-0 ETH
Mar 27 · remix.ethereum.org
-0 ETH
Update
-0 ETH
Mar 27 · remix.ethereum.org
-0 ETH

Improvements

- Amount to be split to be decided on-chain
- Use a separate reward coin
- Design decentralized repository

Thank You