Stake Finance DAO

Software Specifications

(1) Introduction:

In today's booming blockchain era, new technologies and new ideas emerge in an endless stream. The emergence of DAO is a new trend of social development under the wave of blockchain technology. At the same time, as more and more public chains adopt the POS consensus mechanism, the pledge of native tokens has also become a rigid need. Therefore, the combination of DAO and POS pledge is inevitable , we can call it the Stake Finance DAO or the StaFi DAO.

In order to facilitate developers and users to better understand the design ideas and concepts of "Stake Finance DAO", this software was proposed after preliminary feasibility study and demand analysis, and in-depth discussion and analysis with programmers. Specification sheet.

This document is a phased summary of the work of a system analyst and provides a reference for project managers, designers, and developers.

(2) Task overview:

Mission goal: To establish a DAO that can self-govern the community and provide staking liquidity for nodes, which can bring higher user stickiness to the moonbeam ecosystem, bring more lucrative returns to moonbeam contributors, and bring benefits to moonbeam nodes. It will bring more liquidity, break the monopoly position of nodes, bring a more favorable balance to the competition between nodes, and make the ecological development more active and lasting.

Operating environment: On-chain smart contracts run on the EVM of moonbeam network and moonriver network, on-chain transactions are based on ZENLink's SWAP; off-chain timers trigger on-chain interfaces based on JAVA Quartz.

Conditional restrictions: limited by the team's knowledge level, technical ability and development experience; limited by the interface opening ability of moonbeam precompiled files; limited by EVM's support for triggering timing events.

(3) Data description:

Data storage: All data is stored on the moonbeam chain, and only a small amount of auxiliary data for interface triggering is stored off-chain.

Data dictionary: time limit [30 days, 60 days, 180 days, 360 days, 720 days]; fee ratio [20, 30]; demand type [client, collector, validator]; ticket type [ECToken, OCToken] .

System interface: refer to the precompiled interface file StakingInterface.sol, and refer to the audited interface file Openzeppelin.

(4) Functional requirements: mainly include functional division and functional description.

Function division: Pool module, Faucet module, governance module, SWAP.

Function description:

1. Pool module

Open Pool: When the startup conditions are met, the pledge of the Pool is enabled;

User pledge: The user pledges Token (MOVR or GLMR) and mints the ownership certificate OCToken;

Airdrop rights: If the airdrop conditions are met, the rights certificate ECToken is minted and airdropped to the corresponding users;

Distribution income: distribute the pledge income obtained by the collector leasing the Token;

Redemption pledge: The user applies to redeem the pledged Token in the Pool

2. Faucet module

Set the collector: Faucet of the principal type needs to set the collector of the pledge, and the Faucet of the collector type does not need to be set, and the default is the contract address;

Leasing Token: Use ECToken to obtain Token from the Pool and pledge it to the collector, and set the lease time limit;

Recycling Token: After the lease expires, the leased Token will be automatically recycled and returned to the Pool.

3. Governance module

Publishing proposals: Users who own OCToken and meet a certain time limit can publish DAO governance proposals;

Governance voting: Users who own OCToken and meet a certain time limit use ECToken to vote on proposals.

4. SWAP

Token transaction: Users and collectors can exchange ECToken and USDT.

(5) Performance requirements:

The performance of smart contracts is mainly restricted by moonbeam's own TPS. Considering optimization and governance, conditions such as time limit and number of people are designed as variable parameters for governance adjustment. In order to ensure the smooth execution of the contract and the maximum degree of decentralization, the off-chain mechanism is specially set to trigger specific contract interfaces, such as the dividend interface and the startup interface.

(6) Operational requirements:

User interface: DAPP is developed through web3js based on the StaFi DAO smart contract.

Troubleshooting: Contract faults mainly come from contract loopholes, design errors, on-chain congestion and other unknown factors. A set of preprocessing solutions for the above problems are provided to ensure the normal operation and timely processing of contracts.

(7) Other requirements:

Security: Introduce a third-party smart contract auditing agency for contract auditing and testing.

Portability: The smart contract can be applied to other public chains that support EVM.

(8) Milestones and Delivery:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Time node | Project stage | Completed content | Milestone |
| 1 | 2022.2 | Demand research | Demand point analysis  Feasibility Analysis  Future development analysis | Organization conducts needs review |
| 2 | 2022.5 | Design and Development | System Architecture Design  Function module development | Organizing system design review |
| 3 | 2022.7 | Test verification | Alpha testnet deployment  System function test  Test optimization | Organize online review |
|  | 2022.7 | Code audit | Third-party agency audit  Bugfix  Audit optimization | Audit report |
|  | 2022.8 | Official release | Launched on moonriver network  Launched on moonbeam network  Create a pool trading pair |  |
|  | 2023 | Future planning | DAO fundraising and lending |  |