**Antlia Staking - An incentivized staking platform.**

**Introduction**

Our Mission is to increase participation in the securing of proof of stake blockchains and use derivatives to create liquidity.

Proof of Stake Networks are the blockchains of the future due to being exponentially more scalable and environmentally friendly; however, they rely on the active participation of thousands of users all over the world to remain decentralised and secure.

While users are rewarded an average of 6-15% per year for participating in proof of stake, these returns do not compare to other opportunities, specifically in the DeFi sector. In addition, funds in DeFi are liquid, whereas staked assets have a lock period which can range from days to years. This creates the issue of why users would participate in staking networks when rewards on DeFi protocols can be n times higher with the added benefit of being liquid.

To solve this, Antlia Stake App “StakeFlow” has taken a creative approach and has incorporated an incentive model using a native governance token (ANAGOV), paired with the issuance of synthetic assets.

It works simply. Users stake ATOM, DOT, or ETH on the Antlia Staking platform. StakeFlow first phase enroll with Ethereum staking.

In return, receive synthetic assets such as ETH2 (ANAETH2). Thus, solving the liquidity crisis of staked ETH being locked and inaccessible for ~2 years in the case of Ethereum.

Synthetic assets can be traded on UniSwap or AntliaSWAP, used on lending platforms to receive interest or to borrow against, or providing liquidity to UniSwap Pools or Antlia DeFi Pools such as AnAETH/ETH and receive ANAGOV in return for staking their LP tokens. Thus creating much added value out of a locked staking token.

Instead of traditional staking models where users just receive staking rewards, with Antlia Staking platform “StakeFlow” users receive staking rewards + ANAGOV rewards/LendAPY + Liquidity Multiplier.

As we saw with the rapid growth of users in DeFi, we hope this first iteration of a staking incentive model triggers the explosion of proof of stake.

**Token**

Utility: Manage ANAGOV tokens, networks, rewards. Also a form of rebate (i.e if validator fee is 8% and APY from ANAGOV is 12%, your validator fee effectively becomes - 4%).

Obtain

Provide Liquidity on specified uniswap pools and stake LP tokens to receive ANAGOV

Fee

A 10% platform fee is deducted when users withdraw ANAGOV rewards. This is sent to Antlia StakeFlow smart contract and used to maintain the platform, conduct security audits, integrate new features and networks, etc. This fee percentage can be changed at any time via on-chain governance as can the recipient address if Antlia StakeFlow team b does not fulfill its obligations.

Economics

10,000,000 Total.

(max supply can or cannot be changed by community, use whatever is already in contract)

ETH-ANAGOV Pool: 1,000 per day for 1,000 days 1m (10%)

ETH-ANAETH2 Pool: 1,000 per day for 1,000 days 1m (10%)

ETH-NAATOM Pool: 1,000 per day for 1,000 days 1m (10%)

ETH-ANADOT Pool: 1,000 per day for 1,000 days 1m (10%)

60% left for future rewards, all controlled by community.

**How it Works**

For ETH and ERC20 Tokens it is straightforward. The Antlia Staking Dapp StakeFlow, sits in between the user and the validators. Users send ETH to the Antlia Staking Dapp smart contract, and it sends ANAETH2 back to the user and the ETH to the deposit contract/validators.

\*\*ideally we would modify this and make it non-custodial. Where we only hold the validator key and users hold withdrawal keys.

To withdraw ETH2, users burn ANAETH2 and then are eligible to withdraw an equivalent amount + rewards. I.e Deposit 10 ETH, receive 10 ANAETH2. Balance is 10.5ETH2 including rewards, burn 1 ANETH2, withdraw 1.05ETH2 (1 + share of reward). \*Note that ETH2 cannot be withdrawn for ~2 years but after that this is how it will work. When ETH2 Phase2 launches and tokens are withdrawable ANETH2 can be redeemed for ETH2 + Staking Rewards.

For non-ETH based networks this involves a cross-chain interaction, which means a set of contracts sit on both networks which communicate via importing/exporting block information to trigger the mint and burn functions on the respective chains.

\*When a user transfers a token to the Antlia staking dapp “StakeFlow”, it becomes controlled by dapp. User redeems it by burning synthetic equivalent. For example, if alice stakes 1 ATOM and then sells her ANAATOM. She can only withdraw the native token if she purchases 1ANATOM again. Conversely, Bob who purchased the NAATOM can redeem it for 1 native ATOM + rewards even though he initially did not stake anything.

\*Think about this. How to split rewards correctly because if we pool them it is not fair to people who were in early because they should be receiving more rewards than those who just joined.

**Governance**

Token Management - Requests to manually distribute an amount or change network fees.

Reward Management - vote for new projects to be added and reward details.

Initial Networks are ETH2, Antlia, Cosmos, Polkadot, BNB. Additional networks: approved via on-chain governance. Token Address, Amount, Time period.

**Technology** (List what we use and how we modified it)

We are not reinventing the wheel here. We are simply trying to help staking compete with DeFi in terms of yield potential without having to rely on complex derivatives. Because we believe that securing proof of stake networks is much more important than providing liquidity to n AMMs.

We are using existing, battle tested code. Our hope is that this vastly reduces bug related risks because a majority of the code we use has existed in the wild for months or even years. We just make some slight modifications and clearly document everything we have updated so it is easily verifiable and audited for security purposes.

* Compound (Gov + Lending)
* Uniswap (Gov/Community Fund/Pool Token Rewards)
* Cream.Finance (ETH2)
* Proxy Contracts and Cross-Chain (Cosmos, DOT, etc)
* Peggy to facilitate with ETH-Cosmos transfers
* App.Lunie.io Web wallet and browser extension

**Security**

Audits:

**Roadmap**

Phase 1: GUI + Governance + LP rewards + ETH Staking + antETH2

Phase 1.5: ERC20 Staking + Synthetics + LP Rewards

Phase 2: Lending Synthetics

Phase 3: Cosmos Staking + Synthetics + LP Rewards

Phase 3.5: Any Cosmos-based chain

Phase 4: Polkadot

Phase 5: Continued integrations and rewards for other PoS networks based on community governance

**Detailed Roadmap**

Release 1: A fully functional dapp and UI for staking ETH, receiving Synthetic ETH2, ANAGOV Rewards, and on-chain governance

**ETH Staking + Synthetic**. Cream.finance contract modified for non-custodial

**ANAGOV Reward Token** for staking Uniswap LP (UniSwap contract)

**Governance** some combination of Uniswap/Compound should make this pretty out of the box (must be easy to add new networks, including non-eth, manage fees, reward amounts, new projects)

Users vote to add in rewards for new network (amount and time)

Users vote to withdraw funds from community fund

Users vote to update platform fees or existing rewards

**GUI**

* Lunie.io Repo - Refactor to remove mobile wallets. Just have web wallet and browser extension. <https://github.com/Tokenweb/lunie>
* Rebranded Lunie.io Web Wallet and configure VM on dedicated server to run application.
* Rebranded Lunie Browser Extension and Integration to new Web Wallet

Release 2: ERC20 Staking + Synthetics + LP Rewards

Kyber, The Graph

(Using on-chain governance to add new networks)

Release 3: Lending Synthetics

(Based on Compound)

Then it transitions to non-based chains which causes much more complexity. Cosmos, Polkadot, and beyond.

\*All modifications to codebase must be fully documented clearly and concisely.

**We need to first find out that how ?**

1. **How exactly does cream.finance creth2 contract work? ETH -> ETH Deposit Contract, then CREAM.FInance community governed address owns all the ETH in the contract? When ETH2 Phase 2 goes live in ~2 years how does the redemption (creth2 to eth2) and staking rewards work?**
2. **Launch app in first phase**