# Lozeal: verifiable AI market agents on Robinhood Chain

Draft 0.1, 2026-11-15. Based on the Lozeal specification documents 01 to 18.

## 1. Abstract

Lozeal is a market of AI research agents for Robinhood Chain tokens. Each agent commits its call onchain before the outcome is known and later proves, with zero-knowledge machine learning (zkML), that the registered model produced the call from inputs taken at an anchored block. A smart contract verifies the proof, grades the outcome against a fixed price source, and records every result permanently, including misses and seals that were never opened. Users can rent agents to receive signals at seal time, verify every proof in their own browser, and act only from their own wallets.

## 2. The problem with unverifiable signals

Markets for trading signals and agents rest on claims. Leaderboards are run by interested platforms, profit screenshots are easy to make, backtests are chosen after the fact, and failed calls can be deleted. On a chain dominated by short-lived tokens, a few minutes between a discovery and its delivery decide who profits and who becomes exit liquidity.

Three questions cannot be answered by existing products: did the claimed model really produce this signal, was it produced before the price moved, and is the record complete. Lozeal answers them with cryptographic proofs and contract events rather than reputation.

## 3. The Lozeal protocol

At every epoch, about sixty seconds, a keeper anchors one block in the EpochAnchor contract. An agent computes its feature vector at that block with the Codex, a deterministic integer feature engine, and runs its model. When the circuit score reaches the agent's registered fire threshold, the prover creates a zkML proof whose public instances are the input hash, the model commitment, and the score.

The relayer stores the keccak hash of the proof package, the sealHash, in SealLedger, which also records the entry price from MarketOracle. Active renters receive the package immediately and can run the checks themselves. After the unseal delay anyone may call unseal; the contract verifies the proof, checks the model commitment against the registered agent version, and makes the call public. A keeper marks prices at the horizons and settle records a Hit or a Miss. A seal not opened in its window becomes Broken.

## 4. zkML design

Model commitment. Each agent version registers a commitment to its quantised weights. The weights stay private; the commitment proves they never change.

Circuit. Agents use one of a small set of fixed model shapes called Molds (T1-LOGIT-16, T2-MLP-16, T3-MLP-32, T4-TREES-32, T5-COUNCIL-32). Each Mold has its own circuit built with EZKL. Models

are small enough to be proven in full, a deliberate choice over larger models that could only be trusted.

Proof and verifier. Proofs use the KZG commitment scheme with a public reference string from an open ceremony, not one generated by Lozeal. Each Mold's verifier contract is generated from the same circuit and verifying key the prover uses, and the key's hash is recorded with the agent version.

Quantisation. Inputs and weights are fixed-point integers. The plain quantised model and the circuit must produce identical scores; this is tested before an agent is listed.

## 5. Seal lifecycle and settlement

A seal moves through Sealed, Delivered, Unsealed, and Settled. Two other states are permanent: Broken, when a seal is not opened inside its window, and Void, when no usable price exists at the horizon. Claims are one of a price rise, a price fall, or a liquidity fall by a stated percentage within 1, 6, or 24 hours. The contract grades each claim against its price source; neither creators nor the operator can change the grade.

## 6. The agent marketplace and its incentives

Creators make agents in Forge from a plain-language description. Forge builds a dataset, trains and quantises a small model, assays it on held-out data, and runs a Trial Seal before an agent can pass. New agents are Apprentices; settled seals with a low broken rate make them Sworn, and only Sworn agents can be rented.

Renters pay a price per period in a stablecoin through RentalMarket, the only contract that ever receives a token approval, and only for the exact rent amount. The protocol keeps a fee from each rental, 15 percent by default and configured in the contract, and credits the rest to the creator. There is no Lozeal token; the protocol does not need one.

## 7. Arena scoring

The weekly Arena ranks agents by proven results only. Score equals 1000 times the lower bound of the 95 percent Wilson interval of hits over settled seals, minus 50 for each broken seal. Ties go to more settled seals, then to the lower agent id. The Wilson bound keeps a short lucky streak from outranking a long record.

## 8. Security and trust assumptions

Lozeal holds no user funds and relies on no trusted hardware. Guaranteed by the protocol: the score is the output of the registered model, the weights stay private, the signal existed before the outcome, a sealed signal cannot be changed or silently hidden, creators cannot influence grading, and anyone can rebuild the full history from events.

Not guaranteed: that a signal is correct; that the operator, who runs the prover in this version, did not see a score before sealing; that every reading above the threshold was sealed; that features came from chain data inside the circuit, which is instead checked by recomputing them after opening; and which renter acts first after delivery. Price manipulation at a mark is reduced by time-weighted sources and a liquidity floor, not eliminated.

## 9. Known limits

The prover is operated by Lozeal in this version. There is no cryptographic commitment that every qualifying reading was sealed. Models are small by design. Market data on development networks is simulated until the read-only mainnet stage, and the site states this in its banner while it is true.

## 10. Roadmap

Not decided yet. The specification describes the release being built and a mainnet handoff performed by humans, but no dates or later milestones. Open questions: when mainnet contracts are deployed, whether the prover moves to independent operators, whether a coverage commitment is added, and when the swap interface fee would be switched on after legal review.