FRAXT

PvE Meme Markets

Clearer launches, anti-vamp ticker cooldowns, 0x…101 contract addresses, creator fees, and staked ETH rewards.

Whitepaper · v1.3 · July 2026 · Fraxt.io · Built on Robinhood Chain

FRAXT

PvE Meme Markets, Clearer Launches, and Staked Platform Fees

Whitepaper · v1.3 · July 2026

FRAXT is a PvE launchpad for meme markets on Robinhood Chain. It lets creators launch a token, open a live ETH trading market, and publish the market to a discoverable token page from one wallet flow. The design goal is simple: keep meme markets fast, while making launches easier to verify, harder to copy instantly, and clearer for normal users to understand.

Under the hood, the current factory deploys each token and creates a Uniswap V3 market on Robinhood Chain. Users do not need to understand Uniswap internals to use the product: in the interface, a launch means the token receives a live trading market. The deeper contract details are included in this paper for creators, integrators, and reviewers who want to inspect how the system works.

FRAXT launches use several public signals: ticker cooldowns, a required 0x…101 token-address suffix, visible creator records, token pages, and explicit fee routing. Trading fees are split at the factory level between the launch creator, FRAXT stakers, and the platform treasury.

Fraxt.so · built on Robinhood Chain · PvE launches · anti-vamp ticker cooldowns · 0x…101 token addresses

Important Notice

This whitepaper is informational only. It is not an offer, solicitation, recommendation, or invitation to buy, sell, stake, or hold any token, security, or financial instrument. It is not investment, legal, tax, accounting, or financial advice.

Tokens launched through FRAXT are speculative, volatile, and high risk. They may lose value, fail to find liquidity, or become impossible to sell. Staking FRAXT does not guarantee any return, yield, distribution, profit, ownership, governance right, redemption right, or equity interest. Staking rewards, if any, depend on actual eligible platform activity, smart-contract functionality, market volume, user participation, operational constraints, and applicable law.

The staking model described here is not proof-of-stake validator staking. It is a platform-fee participation mechanism and should be reviewed separately from protocol staking guidance. FRAXT may modify, delay, disable, or replace any mechanic described here if required for security, compliance, operations, or user protection.

Executive Summary

Meme launchpads proved that people want fast markets and simple token creation. They also exposed the problems that appear when speed is the only goal: spammy feeds, copied tickers, unclear launch provenance, hard-to-read liquidity, and fee models that users cannot easily inspect.

FRAXT keeps the speed and changes the default game. It is designed as a PvE launchpad: creators should compete on ideas, timing, culture, execution, and community, not on who can copy a ticker fastest or hide inside platform clutter.

The product is built around six principles:

1. **Fast token launch.** A creator enters a name, ticker, logo, website, socials, and description, then launches through the same public factory path as every other creator.
2. **Live trading from launch.** The factory opens a live ETH market on Robinhood Chain. The current implementation uses Uniswap V3, but the user-facing idea is simpler: launch, inspect, trade.
3. **Visible launch identity.** The factory requires token contract addresses to end in 0x...101, and the app shows the creator, token address, market address, and launch records.
4. **Anti-vamp ticker cooldowns.** Recent tickers receive a short protection window so copycats cannot immediately reuse the exact same attention signal.
5. **Creator fee participation.** Collected V3 LP fees are claimable through the factory, and the creator share is paid to the creator address stored by the factory.
6. **Staked platform fees.** The configured FRAXT staking rewards contract receives the staker share of eligible ETH revenue and lets stakers claim ETH rewards pro rata.

The goal is not to make meme markets slow, permissioned, or overbuilt. The goal is to make the launch path fast, legible, harder to fake, and less hostile to creators and first-time traders.

PvE Launchpad Model

Many launch environments become PvP by default. Creators spend too much energy fighting platform spam, ticker clones, and confusing launch surfaces. Traders are forced to guess which token is real, where the market lives, and who receives the fees.

FRAXT treats those problems as product-design problems. The platform cannot remove market risk, bad tokens, or competition for attention, but it can reduce avoidable friction:

- **Less spam pressure.** Launches use a consistent wallet-submitted flow with required metadata, token pages, and contract records.
- **Less ticker vamping.** Same-ticker cooldowns slow immediate copycat launches and give the first token a short window to establish provenance.
- **Clearer launch identity.** The 0x...101 suffix, factory address, token page, creator wallet, and market address give users multiple signals to inspect.
- **Visible reward routing.** Creator fees, staker rewards, and treasury routing are defined in the fee model instead of hidden behind platform discretion.

- **Seamless trading.** A launch opens a live market directly, so users can discover, inspect, and trade from the same surface.

This is the PvE thesis: FRAXT should make the hard part launching something people care about, not surviving platform clutter, ticker theft, confusing markets, or invisible fee mechanics.

1. Protocol Snapshot

This section is the quick-reference version of the paper.

Area	Current FRAXT design
Network	Robinhood Chain
Launch model	Wallet-submitted token deployment and direct Uniswap V3 market creation
Launch philosophy	PvE launchpad: reduce spam, ticker vamping, and hidden routing while preserving fast markets
Token supply	Fixed 1,000,000,000 token supply per launch
Initial creator allocation	0% from the factory launch
Market pair	New token / Robinhood Chain WETH
User-facing market language	A launch opens a live ETH trading market
Initial liquidity style	Single-sided token liquidity seeded into Uniswap V3
LP position	V3 LP NFT held by the factory
Token address signature	Contract address must end in 0x...101
Creator fees	Creator receives 50% of collected V3 LP fees
Staking rewards	Stakers receive the configured ETH rewards share through the staking contract
Treasury	Treasury receives the 30% platform share for operations, infrastructure, ecosystem support, campaigns, grants, and growth
Custody model	Users submit wallet transactions; FRAXT should not custody user wallets or trading assets
Main risk posture	Tokens are volatile, contracts can fail, and rewards are variable

Users should be able to verify the system by checking:

- the active factory address;
- the token contract address and 101 suffix;
- the market / pool address;
- the factory-held LP NFT id;
- the creator address stored by the factory;
- the staking rewards contract;
- the treasury address;
- the ticker cooldown status for same-ticker launches;
- the transaction that created the launch.

2. Why FRAXT Exists

The first generation of meme launchpads made token deployment feel instant. That mattered. It lowered the cost of experimentation and let internet culture become a market in minutes.

But the same speed created predictable problems:

- **Weak provenance.** Users often cannot tell which token was launched through the intended platform path.
- **Ticker vamping.** A token that gets attention can be copied quickly with the same ticker or a confusingly similar one.
- **Spam pressure.** Low-effort launches can crowd the feed and make discovery harder for creators and traders.
- **Opaque market setup.** Launch pages can hide where trading happens, who controls the launch position, and what happens after the first trades.
- **Revenue misalignment.** Platforms capture activity while creators and committed users often have no clear fee participation model.

FRAXT is built around a different default: keep the launch fast, but make the rules visible in the interface and in the contracts. The platform should feel less like a fight against platform clutter and more like a clean environment where creators launch, users inspect, and markets decide.

3. Product Flow

FRAXT is a token launch and market interface on Robinhood Chain.

In the base flow, a creator chooses a token name, ticker, logo, website, socials, and description. The app checks ticker availability, prepares a vanity launch salt, calls the factory, deploys the token, creates the market, records launch metadata, and sends users to a token page where they can inspect the token address, market address, creator wallet, chart, trades, holders, and contract links.

The product is designed for repeated use:

- deployment happens through the same public interface for everyone;
- same-ticker cooldowns reduce immediate copycat confusion;
- token pages show contract addresses, market data, age, and creator information;
- trading activity is displayed through the FRAXT interface;
- launch rules are enforced in software rather than hidden in social convention;
- contract-address suffixes make official launches easier to recognize at a glance.

4. Live Market Creation

For normal users, the important point is simple: launching on FRAXT opens a live ETH trading market on Robinhood Chain.

For technical reviewers, the current Robinhood Chain factory does this by creating and initializing a Uniswap V3 pool. When a launch is created, the factory:

1. deploys a fixed-supply ERC-20 token with CREATE2;
2. rejects the launch unless the deployed token address ends in 0x...101;
3. orders the new token against Robinhood Chain WETH;
4. calls the Uniswap V3 position manager to create and initialize the pool if needed;
5. seeds the full token supply as single-sided V3 liquidity;
6. stores the LP NFT token id in lpTokenIdForToken;
7. maps token, market, and creator addresses for later lookup;
8. routes the deployment fee between staking rewards and treasury.

The initial launch design uses:

Parameter	Current value
Chain	Robinhood Chain
Pair asset	WETH / native ETH wrapper
V3 fee tier	1%
Token supply	1,000,000,000 tokens
Creator launch allocation	0%
Starting market cap setting	2 ETH
Graduation tracking threshold	4.2 ETH paired principal
Deployment fee	0.005 ETH

The factory stores the LP NFT. Creators do not receive an initial token allocation from the launch factory. Their primary on-chain participation is the creator share of collected V3 LP fees.

5. The 0x...101 Contract-Address Rule

FRAXT launches use a visible vanity rule: token contract addresses must end in 101.

The app mines a CREATE2 salt before launch. The factory independently verifies the final token address with VanityLib.endsInGit(address). The check masks the final 12 bits of the address and requires them to equal 0x101.

In plain English:

- a normal random launch salt is not enough;
- the address must end with the three hex characters 101;
- if it does not, the factory reverts with BadVanity;
- users can check the suffix directly in a block explorer.

The suffix is not a security guarantee by itself. It is a simple public signal that the token passed the FRAXT factory's vanity-address rule. Users should still inspect the factory address, token page, creator wallet, market address, and transaction history before trading.

6. Ticker Cooldown Protection

Ticker protection is the anti-vamp rule in the application layer and one of the main PvE design choices.

Every ticker starts with a minimum cooldown after launch. As a token gains traction, the cooldown increases. The purpose is simple: the first token to establish attention around a ticker should get a short protection window before another creator can reuse that exact ticker.

The cooldown is based on the previous launch's strongest observed market-cap and phase signal.

Previous launch state	Same-ticker cooldown
Peak market cap under $10,000	1 hour
Peak market cap at least $10,000 while still in bonding or early market phase	2 hours
Post-bonding, graduated, or established market phase	4 hours
Peak market cap at least $250,000	24 hours

The rule normalizes tickers before comparison. For example, $FRAXT, fraxt, and FR-AXT are treated as the same ticker.

The cooldown is not meant to permanently reserve tickers. It is meant to reduce immediate confusion. Users should still inspect contract addresses, token pages, creator information, and market history before trading.

The cooldown is also an anti-spam signal. It does not prevent all copycats, but it makes the cheapest form of vamping less immediate and gives users more time to recognize the first launch tied to a ticker.

7. Fee Routing

The Robinhood V3 factory routes fees through explicit on-chain functions.

Deployment Fee

Each launch requires a 0.005 ETH deployment fee.

Destination	Share
Staking rewards contract	20%
Platform treasury	80%

The staking portion is sent to the configured staking rewards contract as native ETH. The treasury portion is sent to the configured platform treasury.

Collected V3 LP Fees

LP fees are collected by calling claimV3Fees(tokenAddress) on the factory. The function collects fees from the factory-held V3 LP NFT and pays the configured recipients.

For paired-token fees, meaning WETH fees generated by the Robinhood Chain ETH market:

Destination	Share of paired-token LP fees

Creator	50%
FRAXT staking rewards	20%
Platform treasury	30%

The factory unwraps WETH allocated to staking and forwards it as native ETH to the staking rewards contract.

For non-paired token fees, meaning fees denominated in the launched token itself, the current factory sends:

Destination	Share of launched-token LP fees
Creator	50%
Platform treasury	50%
Staking rewards	0%

This distinction exists because the staking rewards contract pays ETH rewards. Only paired-token revenue can be cleanly converted into native ETH by the current factory path.

8. Creator Claims

Creators claim V3 LP fees through the factory, not through a legacy market contract.

The current claim path is:

1. the app identifies the launch and token address;
2. the app simulates claimV3Fees(tokenAddress) against the active factory;
3. if fees exist, the connected wallet sends the claim transaction;
4. the factory collects from the LP NFT;
5. the creator receives the creator share directly;
6. the staking and treasury shares are routed by the factory.

The claim function may be called by any wallet, but the creator share is paid to the creator address stored by the factory. This means a third party can help trigger fee collection, but cannot redirect the creator payout.

Legacy claim functions such as claimFeesAsUSDC exist only for older market contracts and should not be treated as the current V3 launch path.

9. FRAXT Staking Rewards

The current staking contract is a continuous pro-rata ETH rewards contract.

FRAXT holders stake the FRAXT token. Platform rewards are received as native ETH through notifyRewardAmount or direct ETH receipt. The contract tracks:

- total FRAXT staked;
- each wallet's staked balance;
- cumulative ETH received by the rewards contract;
- cumulative ETH paid out;

- claimable ETH per account through earned(account);
- current pending rewards;
- unallocated rewards received before anyone is staked.

When rewards arrive while FRAXT is staked, rewards accrue according to each wallet's share of total staked FRAXT. A wallet with 10% of staked FRAXT receives 10% of newly allocated ETH rewards for that accounting period.

If rewards arrive when no FRAXT is staked, they become unallocatedRewards. The treasury can sweep those unallocated rewards. This avoids silently assigning pre-stake rewards to the first wallet that stakes later.

The staking interface should describe this as claimable ETH rewards from platform activity, not fixed APY, not validator staking, and not a guaranteed return.

10. Eligible Platform Revenue

FRAXT targets routing visible platform revenue back to the people creating and supporting market activity.

The current on-chain V3 factory routes paired-token LP fees as:

- 50% creator share;
- 20% staking rewards share;
- 30% platform treasury share.

In simple terms: 70% of eligible paired-token trading-fee revenue goes to creators and FRAXT stakers. The remaining 30% goes to the platform treasury.

The treasury funds operations, infrastructure, ecosystem support, grants, campaigns, creator support, safety work, and growth. FRAXT should report treasury usage transparently, but the current contract model keeps treasury as one 30% platform bucket.

For the automated system, **eligible platform revenue** means on-chain protocol revenue actually received by the FRAXT fee contracts and marked for platform allocation after excluding amounts that do not belong to the platform.

Excluded amounts may include:

- creator fees or creator rewards;
- user refunds;
- gas reimbursements or relayer costs;
- third-party processing fees;
- taxes, chargebacks, or legally restricted funds;
- promotional budgets, grants, or non-recurring discretionary transfers;
- revenue streams not yet connected to the on-chain factory, staking, or reporting system.

This narrow definition is intentional. It makes platform revenue easier to audit. If revenue is not visible to the factory, staking rewards contract, treasury wallet, or disclosed reporting process, it should not be counted as automatically distributable staking or treasury revenue.

11. Treasury and Ecosystem Support

The 30% treasury share is meant to keep the platform operating and support projects that make FRAXT more valuable: strong memes, original culture, transparent launches, active communities, and creators who continue building after deployment.

Treasury-funded ecosystem support may include:

- homepage or category placement;
- community competitions and prize pools;
- creator grants;
- trading campaigns or fee rebates;
- social amplification;
- matching reward programs;
- liquidity support where safe, disclosed, and operationally appropriate.

Support should be based on visible criteria rather than insider preference. Factors can include meme quality, community activity, holder distribution, organic volume, creator consistency, low scam signals, no obvious wash trading, no impersonation, and no ticker abuse.

12. User Experience

FRAXT should make the rules obvious.

A creator should be able to see:

- token name, ticker, image, website, and socials before launch;
- whether the ticker is available or cooling down;
- whether the vanity 101 address salt is ready;
- the deployment fee;
- the token address and market address after launch;
- creator fee claim status;
- market activity and chart data.

A staker should be able to see:

- FRAXT wallet balance;
- FRAXT staked balance;
- total FRAXT staked;
- claimable ETH;
- ETH rewards paid;
- total platform revenue received by the staking contract.

The interface should avoid presenting staking as fixed APY unless any number shown is clearly labeled as an estimate based on recent activity. A cleaner default is: "Stake FRAXT to earn ETH from platform launch and trading revenue. More staked FRAXT means a larger share."

For first-time traders, the interface should avoid unnecessary protocol jargon at the point of action. The user-facing flow should say that a launch opens a live trading market, while deeper documents can explain that the current implementation uses Uniswap V3.

13. Operational Commitments

Launchpad documents should make explicit what the platform will and will not do.

FRAXT's intended operating commitments are:

- keep launch creation wallet-submitted and inspectable;
- keep the active factory address visible in the interface;
- keep the whitepaper, public contract addresses, and risk notices aligned with the deployed contracts;
- avoid describing staking as fixed yield or validator staking;
- publish material changes to fee routing, factory addresses, staking contracts, or treasury addresses;
- preserve creator-claim access for launches created through FRAXT;
- make paused or disabled functionality obvious in the interface instead of hiding it behind failed transactions.

These commitments are product and disclosure standards. They do not remove smart-contract, market, legal, or operational risk.

14. Risk Framework

FRAXT should be clear about risk from the start.

Market risk. Tokens launched through FRAXT can be volatile and may lose most or all value.

Liquidity risk. A token may become difficult or impossible to sell.

Smart-contract risk. Factory, token, staking, fee routing, and app contracts may contain bugs or vulnerabilities.

Revenue variability. Staking rewards depend on platform activity and eligible revenue. Low platform activity can mean low or zero rewards.

Routing risk. Creator, staking, and treasury splits are enforced by the current factory for claimed paired-token fees. Treasury spending is operational and requires reporting.

Regulatory risk. Platform fee participation can raise legal and regulatory questions. FRAXT may modify, pause, or discontinue staking or distributions if needed.

Operational risk. Early versions may require keeper actions, contract redeployments, app updates, treasury reporting, or emergency controls to protect users.

The right standard is not "risk-free." The right standard is visible rules, transparent accounting, and honest disclosure.

15. Roadmap

Phase 1: Robinhood Chain Launchpad

- Deploy the FRAXT launch interface.
- Enable token creation through the direct V3 factory.
- Enforce exact ticker cooldown protection in the app.
- Enforce 0x...101 token address suffixes in the factory.
- Publish token pages with clear contract and market data.

Phase 2: Platform Token and Staking

- Deploy the FRAXT platform token.
- Deploy the FRAXT staking rewards contract.
- Publish FRAXT token, staking rewards, treasury, factory, and V3 position-manager addresses.
- Route paired-token platform rewards to staking as native ETH.

Phase 3: Reporting and Treasury Transparency

- Publish fee, rewards, creator-claim, staking, and treasury reporting.
- Report treasury-funded ecosystem support, campaigns, grants, and recipients.
- Add interface analytics for total staked, claimable ETH, historical claims, reward inflows, and treasury support.

Phase 4: Hardening

- Continue test coverage for factory deploys, V3 claims, staking rewards, wallet flows, and ticker cooldowns.
- Review contracts and deployment scripts before each production redeploy.
- Add monitoring for failed claims, stuck rewards, unexpected treasury flows, and stale market data.
- Consider additional revenue sources only when they are auditable and disclosed.

16. Conclusion

FRAXT is a launchpad with rules users can understand.

It keeps token creation fast. It opens live Robinhood Chain ETH markets. It enforces a visible 0x...101 token-address suffix. It slows down same-ticker vamping. It routes creator fees, staking rewards, and treasury allocations through the factory. It gives users a clear surface for trading, claiming, staking, and inspecting what happened on-chain.

The thesis is simple: meme markets should be fun, but they should not be careless. A PvE launchpad should reduce spam, make launch identity easier to verify, keep trading seamless, and make creator participation, staker rewards, and treasury support visible enough to inspect.

Appendix A: Ticker Cooldown Formula

Ticker comparison is normalized before applying cooldowns:

1. Trim whitespace.
2. Remove a leading $.
3. Convert to uppercase.
4. Remove non-alphanumeric characters.

The cooldown is calculated from the previous same-ticker launch:

Condition	Cooldown
Peak market cap under $10,000	1 hour
Peak market cap at least $10,000 while still in bonding or early market phase	2 hours
Post-bonding, graduated, or established market phase	4 hours
Peak market cap at least $250,000	24 hours

If multiple same-ticker launches are inside the lookback window, the active lock with the latest expiration controls availability.

Appendix B: Contract Constants and Formulas

Vanity address

addressLow12Bits = uint160(tokenAddress) & 0xFFF

validFraxtTokenAddress = addressLow12Bits == 0x101

Deployment fee

deployFee = 0.005 ETH

stakingDeployFee = deployFee * 20%

treasuryDeployFee = deployFee * 80%

Paired-token V3 LP fees

creatorShare = pairedTokenFees * 50%

stakingShare = pairedTokenFees * 20%

treasuryShare = pairedTokenFees * 30%

Launched-token V3 LP fees

creatorShare = launchedTokenFees * 50%

treasuryShare = launchedTokenFees * 50%

stakingShare = 0

Staking rewards

rewardPerToken += pendingRewards / totalStaked

walletReward = walletStake * (rewardPerToken - walletRewardPerTokenPaid) + storedWalletRewards

If totalStaked is zero when rewards arrive, those rewards become unallocatedRewards rather than being assigned to a future first staker.

Appendix C: Current Contract Surfaces

The production system should publish the deployed addresses for:

- FRAXT token;
- FRAXT staking rewards contract;
- Robinhood Chain V3 launch factory;
- platform treasury;
- Robinhood Chain WETH;
- Robinhood Chain Uniswap V3 position manager.

The main factory functions users and integrators should understand are:

- createHoodLaunch(...) for token and market creation;
- claimV3Fees(tokenAddress) for creator, staking, and treasury fee routing;
- graduationStatus(tokenAddress) for paired-principal tracking;
- lpTokenIdForToken(tokenAddress) for the factory-held V3 LP NFT id;
- creatorForToken(tokenAddress) for creator payout lookup.

The main staking functions users and integrators should understand are:

- stake(amount);
- withdraw(amount);
- earned(account);
- getReward();
- exit();
- notifyRewardAmount(amount).

Appendix D: Source Notes

1. This paper describes the current Robinhood Chain V3 factory and FRAXT staking rewards architecture in the repository as of July 2026.
2. The staking model described in this paper is a platform-fee participation design, not proof-of-stake validator staking. It requires separate legal and regulatory review before launch.