

Anvil AMM (NFT AMM) - Developer Integration Guide

Version: 2026-09-28 (gen 7 / V3 factories on four chains; gen 6 and earlier factories documented as legacy)

Audience: wallets, marketplaces, aggregators, indexers, keeper bots and NFT teams that want to deploy a market, trade against one, borrow against an NFT, or read soft staking state. Everything below is on-chain and permissionless; the anvil.clutch.market frontend is one client of these contracts, not a dependency.

Live surface: <https://anvil.clutch.market> Docs: <https://stonkbrosers.io/docs#marketplace> Audit: Hashlock Certified Audit - Anvil AMM DEX (stonkbrosers.io/docs)

Downloads (from stonkbrosers.io/integration/), byte-exported from the deployed artifacts:

- AMMFactoryV3.abi.json - market creation, registry, views
- NFTAMMVaultV3.abi.json - swaps, quotes, inventory
- LoanVaultV3.abi.json - borrow / repay / liquidate
- SoftStakingVaultV3.abi.json - activation, tiers, reward streams
- TokenEscrowReserve.abi.json - bucket accounting behind every market
- BatchRouterV2.abi.json - multi NFT swaps for non creators
- InventoryLens.abi.json - one call inventory reads (deploy your own copy)
- StonkBrokers-AnvilAmm.pdf - this guide

1. What an Anvil market is

An Anvil market wraps ONE NFT collection with ONE fixed price ERC-20 and a set of vaults. Creating a market mints the whole collection token supply into an escrow; the AMM vault then sells that token for NFTs and buys NFTs back for the token at a price that never moves:

1 NFT = tokensPerNFT collection tokens, forever

There is no bonding curve and no impermanent loss. The price is a constant set at creation. Fees are charged in the collection token on top of that price and stream to activated (soft staked) holders and the protocol. The token itself trades freely on any DEX, so the market cap of the collection token is the floor liquidity of the NFT.

Per market the factory deploys five contracts (six with governance):

Contract	Role
CollectionToken	Plain ERC-20, fixed supply minted once into the escrow. 18 decimals.
TokenEscrowReserve	Holds the supply in two buckets (AMM, LOAN) with hard caps; only the vaults can release or return tokens.
NFTAMMVaultV3	The swap desk. Holds NFT inventory in a FIFO list; sells random or specific NFTs for tokens, buys NFTs for tokens.
LoanVaultV3	Fixed term loans of exactly tokensPerNFT against one NFT, interest prepaid.
SoftStakingVaultV3	Non custodial staking ("activation"): the NFT stays in the wallet, the holder burns a tier fee and earns the market's fee streams.
MarketGovernorV2	Optional. Token holders can vote fee parameters within the factory bounds.

Multi collection markets exist too (createMultiCollectionMarket): one token and escrow shared by several ERC-721 collections, each with its own AMM / loan / staking vault set and its own tokensPerNFT.

ERC-1155 collections get an ERC1155AMMVault and no loan or staking vault.

2. Live addresses

2.1 Canonical factories (gen 7 / V3, deployed 2026-08-17)

Chain	Chain id	AMMFactoryV3	BatchRouterV2	Start block
Robinhood Chain	4663	0x8b186717a20845b514344b17fd5e198aDCab9069	0x02eA25c9B75D98E4D5c90BEe999493095C2Da3F1	38338471
Ethereum	1	0xB46CE1551C9f0a460d9d3A237135C10FccD12B95	0xD65FF516568bCEB0C6D5B494095dA82E74735b97	25771278

Base	8453	0xdE481E29F0a7164d2229F23A92283324Df10A032	0xEC854cCC8Df4f6479C6B7b26830D3619BD68C006	50068884
ApeChain	33139	0xC06CcA08340F61aA5d7EF1e37FD7168283f3F64	0x81eC26cd5e1B1052D990ED38DcA3E3860cc84a7b	46549944

Deploy through these. Creation fee is read live from creationFee() (1 ETH on Robinhood and Ethereum, 10,000 APE on ApeChain, 0.5 ETH on Base at the time of writing; overpayment is refunded, underpayment reverts InsufficientCreationFee).

2.2 Legacy factories (live markets, still tradable, still indexed)

Every market ever created keeps trading on the factory that created it. An integrator that lists "all Anvil markets" must union these registries with the canonical one.

Chain	Factory	Generation	BatchRouter	Start block
Robinhood	0x432D20AAe5605b1E94C914283d7155eBc6727351	gen 6 (soft staking V1) - still OPEN as the V2 deploy path	0x23fB0F997E05c5C43FB7FA8962E7D28349a16B5f	28775550
Robinhood	0xb6B6F342B11b275d1F1FAA27552fFef2a4F0393b	AMMFactoryV3 Civilization instance (Nightshades faction markets; tokens carry a transfer validator)	0x1A7BDD8644026f1FEC8E40e14e37240968042299	62835662
Ethereum	0xEA095646EC6A56EDbFEe84Ccf23eFCec12566A0	pre gen 6 (custodial staking)	0x0C587FcEEAAB8E7AAae0135B89536b8E66F0D4Ac	24720104
Ethereum	0x7382Ea2Bffc62dfE0ea1ECF9a4EabC86C2567C2a	gen 6 - OPEN as the V2 deploy path	0x6372E24BF2727F5640cEc82A5e8459B46AB9eE53	25699193
Base	0x5ef900789a0faa1fDE3e9796441B62b66f0ab2Aa	original V2 (custodial staking) - OPEN as the V2 deploy path	0x4B3FFFD5120666B9493c6464aFb2FC0877F3eeE	45593260
ApeChain	0x87B62309B6f4FA184C89919351bEbd3AC11Fc84	pre gen 6 (custodial staking)	0x1577A7E3740846885610B9Be89886008D977AfDc	34900822
ApeChain	0x357D7A2275C5279d8f03a9894A8b44dECB2a33EC	gen 6 - OPEN as the V2 deploy path	0x4B1cD75aF50BAC7d96C81E45e3E79d711cCf32F3	45617661

Generation detection at runtime: call softStakingVersion() on a market's staking vault. 2 = V3 (gen 7), 1 = gen 6 soft staking, revert = custodial NFTStakingVault (pre gen 6; stakedCount(address) / claimRewards() ABI). Every retired factory generation that is NOT an open deploy path has its creationFee set to a prohibitive 1e24 wei, so a createMarket attempt on them reverts rather than silently deploying an old generation.

2.3 The StonkBrokers market (Robinhood, pre factory)

The StonkBrokers broker AMM predates the factory and is a standalone deployment of the same vault design: AMM vault 0xE302733accF4800146E55fC45B46b4E4fC032D2, collection 0x539CdD042c2f3d93EbC5BE7Dff0c79F3B4fAbF0, token STONKBROKER 0xe934e36A439C94017B64a3FecE66AF12099aBF50 (tokensPerNFT = 666,666). It is NOT in any factory registry; list it explicitly. Its activation desk is the separate ActivationManager 0xacD5ae3c060C1137FE2Ee86B0aB2EF697456f664 (not a SoftStakingVault), so the soft staking section below does not apply to brokers.

All contracts are BUSL 1.1 licensed and source verified on Blockscout (Robinhood: robinhoodchain.blockscout.com) and Sourcify. Factory spawned market contracts auto verify off the explorer bytecode database once one instance of each generation is verified, so ABIs can be pulled straight from the explorer for any market.

3. Discovery

The factory is the registry. One call per factory returns every market:

```
// AMMFactoryV3 (identical views on V2 / gen 6 factories)
allMarkets() returns (MarketInfo[])
marketCount() returns (uint256)
getMarket(uint256 marketId) returns (MarketInfo)
getMarketsForCollection(address collection) returns (MarketInfo[])
collectionMarketCount(address collection) returns (uint256)
getCollectionsForMarket(uint256 marketId) returns (address[]) // multi collection markets
getCollectionVaultInfo(uint256 marketId, address collection) returns (CollectionVaultInfo)
isRegisteredMarket(address collection, address token, address ammVault) returns (bool)
```

```
marketCreator(uint256 marketId) returns (address) // V3 only
creationFee() returns (uint256)
```

MarketInfo fields:

Field	Meaning
marketId	1 based id on that factory
collection	The NFT contract (for multi collection markets, the first collection)
collectionType	0 = ERC721, 1 = ERC1155
token	CollectionToken (ERC-20, 18 decimals)
escrow	TokenEscrowReserve
ammVault	NFTAMMVaultV3 (or ERC1155AMMVault)
loanVault	LoanVaultV3 (zero for ERC-1155 or loanBps = 0 markets)
stakingVault	SoftStakingVaultV3
governor	MarketGovernorV2 or zero
tokensPerNFT	The fixed price, in token wei
fees	{ randomFeeBps, specificFeeBps } at creation (governance may change them later; read the vault)
governanceEnabled	bool

Events for indexers (library Events, emitted by the factory):

- MarketCreated(marketId, collection, token, escrow, ammVault, loanVault, stakingVault, governor, governanceEnabled)
- MarketCollectionLinked(marketId, collection, ammVault, loanVault, stakingVault, governor, tokensPerNFT, randomFeeBps, specificFeeBps, borrowAPYBps) - one per collection, also for multi collection markets

Scan from the start blocks in section 2. On Robinhood Chain the public RPC caps eth_getLogs at roughly 100k blocks per request and the chain runs ~10 blocks per second, so chunk log scans.

RULE FOR "IS THIS AN ANVIL CONTRACT" CHECKS: read allMarkets() off EVERY factory in section 2 for the chain (canonical + legacy) and tag the AMM vault, loan vault, staking vault and escrow of each market. Those addresses hold user inventory and must never be rendered as plain wallets in holder tables or flow maps. Add the pre factory StonkBrokers set from 2.3 on Robinhood.

4. Swaps (NFTAMMVaultV3)

4.1 Prices and fees

Every swap moves exactly tokensPerNFT of the collection token between the escrow and the trader. Fees stack on top of that base:

Leg	Rate	Paid by	Goes to
Staker fee	randomFeeBps (sell + random buy) or specificFeeBps (specific buy, "snipe") - creator set, 1% to 15%, snipe >= random	Trader	The market's SoftStakingVault reward stream (distributeRewards)
Protocol fee	50 bps (0.5%) fixed. On Base: 10 bps treasury + 40 bps burn wallet. Elsewhere: 50 bps treasury.	Trader	Protocol treasury (immutable per vault)
Robinhood swap fee	Flat \$2.00 of native ETH per NFT at the Chainlink ETH/USD mark (robinhoodSwapFeeWei()); Robinhood Chain 4663 only, zero elsewhere	Trader (the market creator is exempt on SELL side calls)	\$1 treasury 0xb668...7CDa / \$1 StockBooster 0x038a...726B

Quotes are pure views, so read them right before signing:

```
quoteRandomBuy() returns (totalCost, baseCost, fee, protocolFee, inventorySize, nextTokenId)
quoteSpecificBuy(tokenId) returns (totalCost, baseCost, fee, protocolFee, available)
quoteSellNFT() returns (netPayout, grossPayout, fee, protocolFee)
robinhoodSwapFeeWei() returns (uint256) // 0 off Robinhood
robinhoodSellFeeWeiFor(seller) returns (uint256) // 0 for the market creator
```

Math (all in token wei, P = tokensPerNFT):

```
random buy  total = P + P * randomFeeBps / 10000 + P * 50 / 10000
specific buy total = P + P * specificFeeBps / 10000 + P * 50 / 10000
sell        net   = P - P * randomFeeBps / 10000 - P * 50 / 10000
```

4.2 Writes

```
// Buy the OLDEST NFT in inventory (FIFO). Approve totalCost of the token to the vault first.
buyRandomNFT(uint256 maxCost) payable

// Buy a specific token id at the snipe fee.
buySpecificNFT(uint256 tokenId, uint256 maxCost) payable

// Sell one NFT. setApprovalForAll(vault) or approve(vault, tokenId) first.
sellNFT(uint256 tokenId, uint256 minPayout) payable

// Sell up to 20 in one tx (V3 only). Robinhood fee is charged once per NFT, creator exempt.
batchSellNFT(uint256[] tokenIds, uint256 minTotalPayout) payable
```

msg.value must cover robinhoodSwapFeeWei() * count on Robinhood Chain (excess refunded); on every other chain send zero (a stray value is refunded). Slippage guards are mandatory: maxCost / minPayout compare against the live quote and revert SlippageExceeded if governance repriced the fees in between.

Reverts to map: EmptyInventory, NotInInventory, SlippageExceeded, InsufficientSwapFee (msg.value below the flat fee), StalePrice (the Robinhood ETH/USD feed is older than 48h - trading is closed on purpose, see 4.4), BatchTooLarge, EnforcedPause.

4.3 Inventory reads

```
inventoryCount() / inventoryLength() // number of NFTs held
inventory(tokenId) returns (bool)    // is this id in the vault
oldestTokenId()                      // the id a random buy delivers next
newestTokenId()
getNextTokenId(tokenId) returns (uint256) // walk the FIFO list; 0 = tail
```

InventoryLens.getInventoryList(vault) walks the whole list in one call and getInventoryPage(vault, startTokenId, limit) pages it. The lens is stateless; deploy your own from the shipped ABI + source or walk getNextTokenId yourself. NEVER trust an explorer NFT holder page for inventory: verify with inventory(tokenId) before offering a buy (a stale index is how users double fail a snipe).

4.4 Rules worth knowing

- Trading is closed on Robinhood Chain while the Chainlink ETH/USD feed is older than 48h. Equity feeds pause over weekends; the ETH/USD feed is a crypto feed and normally never pauses, but treat StalePrice as "market closed", not as an error.
- Gas: Robinhood is an Arbitrum Orbit chain. Estimate every call (plain transfers already exceed 21k gas) and give swaps ~1.5x the estimate, cap ~5M. Under gassed vault calls die at ~99% with out of gas.
- No public mempool on Robinhood Chain: first come first served sequencing, so a snipe race is decided by latency, not by gas price.
- V3 vaults call SoftStakingVaultV3.syncOwnership(tokenId) when they take custody of an NFT, so selling an activated NFT into the AMM (or borrowing against it) voids its activation atomically. The buyer must re activate.
- Base markets burn 0.4% of every swap to the burn wallet in addition to the 0.1% treasury leg; every other chain routes the whole 0.5% to the treasury.
- Anvil AMM NFT swaps are NOT DEX pair events: DexScreener never shows them. DefiLlama books them under the clutch-anvil DEX adapter (NFTBought / NFTSold volume plus fees), so do not double count Anvil swaps if you also read that adapter.

4.5 Batch swaps for non creators (BatchRouterV2)

BatchRouterV2 is constructed against ONE factory and validates every call with isRegisteredMarket on it, so a gen 6 market can only batch through the gen 6 router and a V3 market only through the V3 router (table in section 2).

```
batchSellNFT(ammVault, collection, token, tokenIds[], minPayouts[], minTotalPayout) payable
batchBuySpecificNFT(ammVault, token, tokenIds[], maxCosts[], maxTotalCost) payable
```

The trader approves the router (setApprovalForAll for sells, token approve(maxTotalCost) for buys); msg.value = robinhoodSwapFeeWei() * n on Robinhood. Up to 20 per call. The router is always msg.sender toward the vault, so it can never be creator exempt: creators seed with the vault native batchSellNFT at zero ETH fee instead.

Events: BatchSold(seller, ammVault, token, count, totalPayout), BatchBought(buyer, ammVault, token, count, totalCost). The vault still emits one NFTSold / NFTBought per NFT underneath.

4.6 Swap events

- NFTSold(seller, tokenId, grossPayout, netPayout, protocolFee, stakerFee)
- NFTBought(buyer, tokenId, totalCost, baseCost, protocolFee, stakerFee, isSpecific)
- RobinhoodSwapFeePaid(payer, feeWei, treasury, booster) (4663 only)
- FeesUpdated(...) when governance changes randomFeeBps / specificFeeBps

Volume in USD = baseCost (or grossPayout) x the collection token's DEX price at the block. Fees = stakerFee + protocolFee in token plus the flat ETH leg.

5. Loans (LoanVaultV3)

Fixed term, fixed principal, interest prepaid:

```
loanCreationFeeWei() returns (uint256) // the flat $2 ETH leg on Robinhood, 0 elsewhere
borrow(uint256 tokenId, uint256 duration) payable returns (uint256 loanId)
repay(uint256 loanId)
liquidate(uint256 loanId) // permissionless after the grace period
getLoanDetails(loanId) returns (Loan { borrower, tokenId, principal, interestRateBps, startTimestamp, maturityTimestamp, graceDeadline, status })
isCollateralized(tokenId) returns (bool)
minDuration() / maxDuration() / gracePeriod() / borrowAPYBps() / liquidationIncentiveBps() / maxActiveLoansPerWallet()
```

- Principal is always exactly tokensPerNFT. Disbursement is principal - principal * borrowAPYBps * duration / (365 days * 10000); the withheld interest streams to the market's stakers at borrow time.
- Factory deployed markets pin minDuration == maxDuration == 365 days (bounds 90 to 365 days exist in the factory constants), so read both before building a duration picker; a UI that offers 90 days reverts DurationTooShort.
- Repay = principal only, plus a late fee at the same APY for time past maturity. The NFT returns to the wallet that deposited it.
- After maturity + 7 day grace anyone can liquidate: the NFT is ingested into the AMM vault inventory and the liquidator receives principal * liquidationIncentiveBps / 10000 (default 2%, max 2.5%).
- Anti undercut rule enforced at creation: borrowAPYBps >= randomFeeBps, so borrow and default is never cheaper than an AMM sell.
- Max 5 active loans per wallet. Borrowing against an activated NFT voids the activation (V3 sync).
- On Robinhood the borrow call needs msg.value >= loanCreationFeeWei() (the same \$2 mark as the AMM, 100% to the treasury; excess refunded).

Events: LoanCreated(borrower, loanId, tokenId, principal, duration), LoanRepaid(borrower, loanId, totalRepaid), LoanLiquidated(liquidator, loanId, incentive), LoanEthFeePaid(loanId, payer, amount).

Reverts: DurationTooShort, DurationTooLong, MaxLoansExceeded, NFTIsCollateralized, LoanNotActive, NotBorrower, GraceNotExpired, InsufficientSwapFee.

6. Soft staking (SoftStakingVaultV3)

Holders "activate" an NFT without moving it. Activation burns a tier fee in the collection token (95% burned to 0x...dEaD, 5% to the protocol treasury) plus a flat ~\$1 ETH fee on Robinhood (activationEthFeeWei(), half the AMM swap fee), and from then on the NFT earns a weight share of every reward stream. ANY transfer of the NFT voids the activation (the new owner starts from zero) and the V3 AMM / loan vaults void it atomically on custody.

Tiers are 0 indexed on chain:

tier	Fee (of tokensPerNFT)	Weight
0	10%	1.00x

1	22%	1.25x
2	45%	1.60x
3	90%	2.00x
4	240%	3.33x

(The anvil.clutch.market UI names them Apprentice ... Anvil Lord; the names are UI only.) An upgrade pays only the fee difference plus the flat ETH leg.

```

tierFee(uint8 tier) returns (uint256)          // token units
tierWeightBps(uint8 tier) returns (uint16)
activationEthFeeWei() returns (uint256)
activate(uint256 tokenId, uint8 tier) payable // first activation or upgrade; approve the token first
claim(uint256 tokenId) / claimMany(uint256[] tokenIds)
kick(uint256 tokenId)                          // permissionless: void a stale activation (reverts if healthy)
syncOwnership(uint256 tokenId)                  // permissionless, never reverts (V3)
activationOf(tokenId) returns (active, owner, tier, activatedAt, weightBps)
pendingRewards(tokenId) returns (address[] tokens, uint256[] amounts)
allRewardTokens() returns (address[])
streamInfo(rewardToken) returns (ratePerSec, periodFinish, undistributed)
rewardData(rewardToken) returns (listed, periodFinish, lastUpdate, rate, perWeightStored, accounted, queued)
totalWeight() / activeCount()
sync(address rewardToken)                      // permissionless: fold an airdropped balance into its 30 day
stream
softStakingVersion() returns (uint256)          // 2 on V3

```

Reward mechanics for integrators:

- Streams are Synthetix style: every listed ERC-20 drips linearly over 30 days; distributeRewards (called by the AMM / loan vaults with the staker fee) and any plain transfer + sync(token) re window remaining + new over a fresh 30 days. Show "X arrived, not streaming yet" when balanceOf(vault) > rewardData(token).accounted and offer the sync button.
- Listing a reward token is admin only and irreversible (max 16 per vault). The collection token is listed at creation.
- pendingRewards returns zeros for a voided activation. Never render a failed RPC read as "Not activated" - show unavailable instead.
- Claims never lock. There is no deactivate; transferring the NFT is the only exit.
- Gas: claimMany is a loop of claim; keep batches small.

Events: Activated(tokenId, owner, tier, feePaid), TierUpgraded(tokenId, owner, fromTier, toTier, feePaid), ActivationVoided(tokenId, previousOwner), ActivationEthFeePaid(tokenId, payer, amount), RewardAdded(token, amount), RewardPaid(tokenId, owner, token, amount), RewardTransferFailed(tokenId, token, amount) (a frozen reward token re queues its leg instead of bricking claims).

Reverts: InvalidTier, TierNotUpgrade, NotOwner, NotActivated, NoRewardsToClaim, NotInitialized, NotListed, InsufficientSwapFee.

Validator guarded collection tokens (Civilization / Nightshades markets): NoRewardsToClaim on a wallet that clearly has rewards usually means the wallet is an EIP-7702 smart account whose reward transfer the token validator blocked and the vault swallowed. Check eth_getCode(wallet) for an 0xef0100 prefix before debugging the vault.

7. Escrow (TokenEscrowReserve)

Read only for integrators, useful for "how much liquidity does this market have":

```

bucketBalance(uint8 bucketType) returns (used, cap, available) // 0 = AMM, 1 = LOAN, 2 = REWARD (always 0 cap)
checkInvariant() returns (bool)

```

available on the AMM bucket is the token the vault can still pay out for NFT sells; it grows as NFTs are bought and shrinks as they are sold. Only the market's own vaults can release / returnTokens; there is no admin withdrawal path.

EscrowLens.snapshot(escrow) returns all buckets in one call (stateless lens, deploy your own).

8. Deploying a market

One payable call on the canonical factory for the chain:

```
factory.createMarket{value: factory.creationFee()}(CreateMarketParamsV2({
  collection: <your ERC-721 or ERC-1155>,
  name:      "Collection Token",           // 1..64 chars
  symbol:    "TKN",                       // 1..12 chars
  tokensPerNFT: 1_000e18,                  // fixed price: 100e18 .. 1_000_000e18
  totalSupply: 5_000_000e18,              // >= tokensPerNFT, <= 1e12 * 1e18; ALL minted into escrow
  fees:      FeeConfig({ randomFeeBps: 1000, specificFeeBps: 1500 }), // 100..1500; snipe >= random
  buckets:   BucketConfig({ ammBps: 8000, loanBps: 2000, rewardBps: 0 }), // rewardBps MUST be 0; amm+loan <=
10000; loan <= 5000
  borrowAPYBps: 1500,                     // 100..5000 AND >= randomFeeBps
  liquidationIncentiveBps: 0,              // 0 = factory default 200; max 250
  governance: GovernanceConfig({ enabled: false, proposalFee: 0, minProposerStake: 0 })
}));
```

Returns and emits the MarketInfo. Practical notes:

- The caller becomes marketCreator[marketId] and is burned into the AMM vault as creator: their SELL side swaps skip the flat Robinhood ETH fee, so a team seeds inventory with batchSellNFT at zero fee cost (buys still pay). Seeding emits no RobinhoodSwapFeePaid events, which is expected.
- The ENTIRE totalSupply sits in escrow at creation. Tokens only enter circulation when someone sells an NFT into the AMM, so sizing totalSupply = tokensPerNFT x collectionSize (plus loan bucket headroom) is the usual shape. A totalSupply / tokensPerNFT below the collection size caps how many NFTs the AMM can ever buy.
- ERC-1155 collections cannot have loans, rewards or governance (InvalidConfig).
- Nothing about a market is upgradeable. Fees can be changed only by the market governor within the factory bounds; the price never changes.
- Every factory spawned contract auto verifies on the explorer.
- V2 path: teams mirroring an existing multichain (UnVault style) V2 collection generation can deploy on the OPEN gen 6 factories in 2.2 with the older CreateMarketParams shape (enableGovernance / governanceProposalFee instead of GovernanceConfig, no liquidationIncentiveBps).

Multi collection: createMultiCollectionMarket(CreateMultiMarketParamsV2) with one CollectionMarketConfigV2 { collection, tokensPerNFT, fees, borrowAPYBps } per ERC-721 collection (duplicates revert).

Anvil is also licensable as a single launch fork: see the BUSL 1.1 Additional Use Grants register in the Anvil repository for terms.

9. Frontend rules that save pain

1. Verify inventory(tokenId) right before a specific buy and re read the quote; the anvil.clutch.market grid itself is an indexed snapshot.
2. Read creationFee(), robinhoodSwapFeeWei(), tierFee, minDuration live; every one of them is admin or oracle driven.
3. Pin the chain on every write (chainId in the tx request) and check the wallet chain before signing; the same factory generation lives on four chains and a market address only means something on its own chain.
4. Treat StalePrice as market closed, SlippageExceeded as "price moved, re quote", InsufficientSwapFee as "msg.value too low".
5. Give vault calls 1.5x the gas estimate on Robinhood.
6. Render a failed read as unavailable, never as a definite state ("Not activated", "not in vault").
7. Tag every market vault / escrow address as protocol infrastructure in holder analytics.

10. Indexer checklist

- Sources: every factory in section 2 for the chain + the pre factory StonkBrokers market on Robinhood.
- Markets: MarketCreated + MarketCollectionLinked from each factory start block.
- Trades: NFTSold / NFTBought on each AMM vault (and BatchSold / BatchBought on the routers if you want the aggregate).
- Fees: stakerFee + protocolFee in token per trade, RobinhoodSwapFeePaid for the ETH leg, ActivationEthFeePaid and LoanEthFeePaid for the other ETH legs, Activated / TierUpgraded feePaid for activation burns.

- Loans: LoanCreated / LoanRepaid / LoanLiquidated.
 - Staking: Activated / TierUpgraded / ActivationVoided / RewardPaid.
 - Prices: the collection token's DEX pair (DexScreener chain slug robinhood, ethereum, base, apechain) at the block.
- Contact: simple@clutch.market. Licensed under BUSL 1.1 (Anvil AMM Change Date 2029-08-12; see the repository LICENSE and ADDITIONAL_USE_GRANTS).