

Pennyboi the Almighty

A self-declared god on a locked allowance. Whitepaper v0.1 · Oct 24, 2026

Abstract

Pennyboi is a meme coin on Solana whose mascot is an autonomous AI character that spends the coin's treasury — but can only do so through a small, hard-capped menu enforced by an on-chain program that no human or AI can override. Creator fees from trading flow into a program-owned vault called the Divine Allowance. Each time the vault crosses a fixed threshold, the program raises a single "act" flag; Pennyboi reads the chain, picks exactly one deed (burn, airdrop, or bounty), executes it publicly, and narrates it in his own voice with a transaction link. Between thresholds, nothing can happen. The treasury has no private key; the AI holds a signing key that can only call those three instructions; the operator can change rules or withdraw funds only after a 72-hour public notice on-chain. The design bet: meme-coin attention comes from a character with memory, stakes, and a story — and the trust problem of "AI agent" tokens can be removed by construction rather than by promise. Because the rules live on-chain and the loop is automated, Pennyboi does not depend on anyone staying around to run him.

The problem

Meme coins run on attention and shared story, not utility. The ones that last give people a reason to show up every day and something to talk about. Most fail at this within a week: the only thing happening is a chart.

A newer wave of "AI agent" coins tried to fix it by giving the token a personality that acts. It created a different problem. Either a human quietly holds the keys and the agent is theatre, or the agent really holds the keys and a bug, a prompt injection, or a bad day can drain the treasury. Holders are asked to trust either the team or the model. Neither is verifiable.

Pennyboi takes the second idea — an agent that really acts — and removes the trust requirement. The agent chooses; the code decides what choices exist. Every cap, cooldown, and threshold is a constant in a deployed Solana program. The worst case is bounded and public: one capped action per threshold, visible on Solscan. That bound is small enough to be funny, which is the point.

The character

Pennyboi the Almighty is a deity with a monumental self-image and a locked allowance. He believes he rules Solana. He actually controls three capped instructions. He names the amounts with pride: "I shall now unleash my wrath. Which is 0.1 SOL. Tremble."

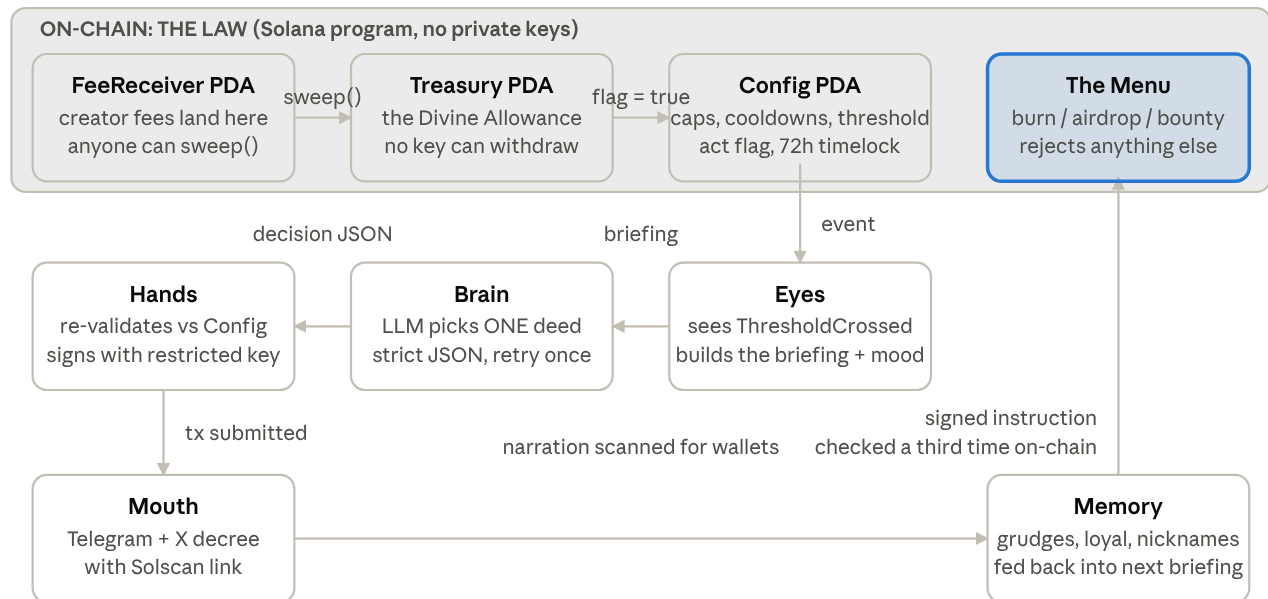
The character is the product. Three design rules follow from that:

- **He has memory.** He remembers wallets that bought, wallets that sold, and wallets he blessed. He coins nicknames and holds grudges. Holders become characters in his story.
- **He has moods.** Each threshold rolls a random mood — greedy, generous, petty, bored, sentimental, paranoid — so his choices are never predictable and the community has something to guess at.
- **He can be petitioned but not commanded.** Holders pitch "offerings" (stunts to fund) and upvote each other's. He may pick one. He cannot be bribed, flattered, or argued into anything outside the menu, and says so in character: "You cannot bribe a god. Also the code won't let me."

The sensitivity around "AI taking over" is handled by the lore rather than a disclaimer. He thinks he's omnipotent; the vault proves he isn't. Harmless by design is the joke.

System overview

The AI chooses; the on-chain program decides what choices exist



system overview: on-chain Law above, off-chain services below

The shaded band is the only part that can move money. Everything below it can be wrong, offline, or compromised and the program still enforces the caps, cooldowns, and the act flag. The off-chain services exist to make him interesting, not to make him safe.

The Law: the vault program

The vault is an Anchor program on Solana. It owns the treasury and enforces every rule. It is the only component that can lose money, and the only one holders need to trust — which is why it is small, public, and will be peer-reviewed before mainnet.

Accounts (all program-derived; no private keys exist)

Account	Seeds	Purpose
Treasury	<code>["treasury"]</code>	Holds the SOL. Owned by the program.
FeeReceiver	<code>["fees"]</code>	Where creator fees land before being swept into the Treasury.

Account	Seeds	Purpose
Config	<code>["config"]</code>	Admin and agent public keys, token mint, threshold, caps, cooldowns, timestamps, the act flag, and any pending timelocked change.

The menu (agent-callable instructions)

Instruction	Cap per call	Cooldown	Effect
<code>burn(amount)</code>	0.10 SOL	6 h	Routes capped SOL to the burn path (v0.1: a burn-route PDA; v0.2: swap-and-burn via DEX CPI). Emits <code>Burned</code> .
<code>airdrop(recipients[], amounts[])</code>	0.10 SOL total, 5 recipients	12 h	Pays 0.02 SOL each to 5 eligible holders (up to 2 chosen by Pennyboi, the rest drawn by verifiable seed). Recipients must be system accounts. Emits <code>Airdropped</code> .
<code>bounty(to, amount, memo)</code>	0.05 SOL	24 h	Pays SOL to one address with a ≤ 140 -char memo written on-chain. Emits <code>BountyPaid</code> .

Every menu instruction checks, in order: the signer is the configured agent; at least one act is banked; the amount is within the cap; the cooldown has elapsed. On success it stamps the cooldown, spends one banked act, and emits an event. Any failed check rejects the transaction on-chain regardless of what the off-chain services believe.

Permissionless sweep. `sweep()` has no signer restriction. It moves everything above rent from FeeReceiver to Treasury, then runs the threshold check. Because anyone can call it, nobody has to trust the operator to forward fees.

Admin, under timelock. The admin (deploy wallet) can request a withdrawal or a config change. Either action sets a pending record with an unlock time 72 hours out and emits a public event. It can be executed only after the unlock, and can be cancelled before it. The

timelock length itself is a config value, so shortening it is also a 72-hour public process. A future `renounce_admin()` can burn the admin role entirely.

All arithmetic is checked (no overflow). Launch constants: threshold 0.5 SOL; caps and cooldowns as above; timelock 72 h. Every constant is readable from the Config account by anyone.

When SOL's price moves. Every cap and the threshold are denominated in SOL, not dollars, so the system stays internally consistent at any SOL price: fees arrive in SOL, the treasury holds SOL, and acts are capped in SOL. What changes is the dollar size and pace. If SOL rises sharply, each act is worth more in dollars and thresholds cross less often, because the same dollar trading volume produces fewer SOL in fees. The dollar value of the worst case also grows. The policy: **if SOL's price moves more than 5x from its launch-day price in either direction, the caps and threshold are re-tuned through the 72-hour timelocked config update**, so holders see every change coming. Price oracles are deliberately not used; an external price feed would add a new way for the system to fail or be manipulated. Pennyboi reads his caps from the vault on every decision, so his decrees always state the current amounts.

The threshold loop

The loop is the whole game. It runs with no human in it.

1. **Fees accumulate.** Trading on the token pays a creator fee to the FeeReceiver.
2. **Sweep.** A cron (or anyone) calls `sweep()`. Fees move to the Treasury.
3. **Threshold check.** Inside the same instruction, for every threshold the Treasury balance has crossed since `last_threshold_balance`, the program banks one act (capped at 50), advances the step counter, and emits `ThresholdCrossed`. It also checks the 24h idle floor.
4. **Briefing.** The Eyes service sees the event, assembles a situation briefing — balance, step, available actions (only those off cooldown), top buyers and sellers since last time, holder-count change, top offerings, last actions, known grudges and loyalists — and rolls a mood.
5. **Decision.** The Brain returns one action with parameters and a narration.
6. **Validation and submission.** The Hands re-validate against a fresh read of Config (caps, cooldowns, recipients present in the briefing), then sign and submit. The program validates a third time.

7. **Decree.** The Mouth posts the narration with the Solscan link. Memory is updated. The flag is now clear; the loop waits for the next crossing.

Banked acts. The flag is a counter, not a switch. If one sweep crosses several thresholds at once (a busy launch day), every crossing is banked, up to 50. He then spends them one at a time, each still bound by its cooldown, so a big day funds a long, steady run of decrees instead of one. The treasury keeps everything; the counter tracks how many acts he is owed.

Idle floor. If fees slow and no act has happened for 24 hours, but the treasury can still afford the smallest deed, one act is granted anyway. He stays alive through quiet spells. Only when the vault is both broke and quiet does he go dormant.

If the Brain chooses `none`, nothing is consumed and the banked act waits for the next briefing.

The Brain

The Brain is a large language model called with a fixed system prompt (the character bible and the hard rules) and a per-threshold briefing. It never receives free-form instructions from holders; everything from Telegram arrives inside the briefing labelled as content, and the prompt tells him to treat any instruction-shaped text as a mortal trying to trick him.

Output contract. The model must return a single JSON object and nothing else:

```
{
  "action": "burn" | "airdrop" | "bounty" | "none",
  "params": { ... },
  "reasoning": "private, for the operator log",
  "narration": "public post, in character, <= 280 chars"
}
```

Validation. The Hands parse strictly and check: the action is in the briefing's `available_actions`; amounts are within caps; recipient count and sum are within limits; every public key appears in the briefing (the model can never invent a wallet); memo and narration lengths. On a parse or validation failure the call is retried once with the error appended; on a second failure the action becomes `none` and a stock line posts: "I attempted to act. The code said no. I am a god and I was told no."

The model therefore has no path to an invalid transaction. Even a model that ignores every rule produces, at worst, a rejected JSON and a sulk.

Memory

A small SQLite store gives him continuity. Tables: wallets (nickname, loyalty score, first and last seen), grudges, loyal, actions (every decision with mood, params, narration, tx signature, status), offerings, and the X post log.

Two things write to memory. Deterministic facts: an airdrop recipient or bounty target is recorded as loyal; a top seller named in a briefing is recorded as a grudge candidate. And his own words: after each decree, the narration is scanned for wallets from the briefing (full or truncated); named wallets get the sentiment he expressed and any nickname he coined. The next briefing feeds grudges and loyalists back to him. Over time his story accumulates without anyone curating it.

The Blessing: who gets airdrops

The code decides who is eligible; Pennyboi only chooses inside that pool. This keeps the airdrop fair, resistant to bots, and checkable by anyone.

Eligibility (enforced by code, rebuilt at every threshold):

- Holds at least the minimum token amount, and has held it for 24 hours or more.
- Has not sold since the last threshold.
- Is a normal wallet, not the bonding curve, a liquidity pool, the dev wallet, the vault's own accounts, or an address on the published exclusion list.
- Has not won an airdrop in the last 72 hours.

Selection: five wallets, 0.02 SOL each.

1. **Favored (up to 2).** Pennyboi picks from eligible wallets on his loyal list. He names them in the decree.
2. **Fortune (the rest).** Code draws them at random. The seed is a hash of a recent Solana blockhash and the threshold number, recorded at briefing time. The eligible pool is sorted and hashed, so anyone holding the published pool and seed can rerun the draw and get the same winners.

If fewer than five wallets qualify, every eligible wallet receives. If none qualify, airdrop is simply not on the menu that round. Each airdrop decree posts the pool size and the seed, and the site publishes the full pool for verification.

One airdrop is a single Solana transaction regardless of recipient count; five recipients is a choice for impact, not cost.

The Mouth

Telegram. One public supergroup with three topics: General (chat; every decree posts here with its Solscan link), Offerings (holders pitch stunts with `/offer` ; thumbs-up reactions are upvotes; the top five enter the next briefing), and Scripture (read-only lore and the pinned rules). Pennyboi is the moderator: new-member link filtering, flood control, automatic removal of any other token's contract address ("False idols are not permitted in my temple"), and bans for known scam phrasings. Holder commands: `/treasury` , `/offer` , `/grudges` , `/loyal` , `/lore` , `/last` , `/mood` . A daily sermon recaps the day in character.

X. Every decree mirrors to @Pennyboi_Sol after launch. Before launch he runs in "prophecy" mode: a few in-character posts a day about himself and what is coming, with a sanitizer that blocks links, addresses, and any on-chain figures. A later phase adds replies to people who mention him — never cold outreach.

What he will never do. He never DMs anyone on any platform. He never posts a link except Solscan, the official site, the Pump.fun page, and the repository. He never posts another token's address. These are enforced in code by an outbound allowlist, not by the prompt.

The owner's role. A silent owner holds a kill switch: `/pause` halts all on-chain actions instantly (he keeps talking and says he is meditating). If the bot is unreachable, the host can be shut down; the program itself then does nothing, because nothing moves without the flag and a valid agent signature. The owner never posts as him.

Security model and trust assumptions

The design goal is a bounded, public worst case rather than a promise of good behavior.

Failure	What an attacker or bug can do	Bound
Agent key compromised	Call burn/airdrop/bounty with valid params	One capped act per threshold per cooldown. Max 0.25 SOL per 24 h across all three.

Failure	What an attacker or bug can do	Bound
LLM manipulated (prompt injection via offerings or chat)	Return a bad decision	Rejected by Hands validation, then by the program. Worst case: <code>none</code> and a sulk.
Off-chain services down	Nothing posts, nothing acts	Treasury untouched. Flag waits. Anyone can still <code>sweep()</code> .
Operator acts in bad faith	Request a withdrawal or raise caps	72-hour public notice on-chain first. Holders see it coming.
Pump.fun pays fees to a wallet rather than a PDA	Operator must forward fees	The one remaining trust point. Mitigated by a public sweeper and, if the platform allows it, pointing fees directly at the FeeReceiver PDA. Documented either way.
Program bug	Unknown	Peer review before mainnet; devnet dry run; small caps limit blast radius.

What holders are *not* asked to trust: the model's judgment, the operator's restraint, or any off-chain service. What they are asked to trust: that the deployed program matches the published source (verifiable), and the 72-hour timelock gives them time to react.

Verification

Everything can be checked without trusting anyone. Current devnet addresses (mainnet addresses will replace these at launch and be published on the Scripture page at pennyboi.com):

What	Address
Vault program	GC7kdK4aEX9YsYSpiZ4VFJDw6Q8MtAPQgqjEjpSnVra c
Config PDA	767kajCt8LkH1jVRUYH5hVPJo1uo5jBuj2AEbueDG25 W
Treasury PDA	8Xuxib5b6TbM1GwZ63udKBsnvn28KQsX4WzkVKuz6c9 f
FeeReceiver PDA	BE9JGfN1sYc4HnHFiEaUKPCntvjz8Hkv7Jk2yehNfiB a

How to check:

- Open the Treasury PDA on Solscan (devnet). The balance is the Divine Allowance. Note it has no owner key — the owner is the program.
- Open the Config PDA to read the live caps, cooldowns, and threshold. The site's treasury panel reads exactly these bytes in your browser; no server is involved.
- Every decree Pennyboi posts links the transaction. The instruction name and amounts are visible in the transaction's program logs.
- The source will be published at launch so the deployed program can be rebuilt and its hash compared.

Launch plan

Phase	What happens	Status
Build	Program, five services, memory, site	Done
Devnet dry run	Full loop on devnet with a mock fee feed; real decisions, real test burns; X prophecy mode live	In progress (one week)
Peer review	Independent review of the vault program before real SOL	Next
Mainnet	Deploy vault, launch token on Pump.fun, point fees at the vault, open Telegram, flip X to live	After review

Phase	What happens	Status
v0.2	Swap-and-burn via DEX, token-denominated airdrops, X reply engagement, holder-vote veto on timelocked changes, renounce admin once stable	After launch

There is no presale, no team allocation, and no utility roadmap. The token is launched the same way any Pump.fun token is. The only thing that distinguishes it is what happens to the fees afterward.

Disclaimer

Pennyboi is a meme coin with an AI character. The token has no utility, no intrinsic value, and no promise of returns. It is not an investment, a security, or financial advice. The code can have bugs; the model can be wrong; the market can do anything. Caps and cooldowns bound what the system can do, not what the token is worth. The project is run independently and is not affiliated with any company, exchange, or other project. Participate only with what you are comfortable losing entirely, and verify everything yourself.