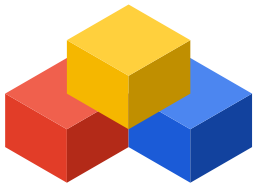

WHITEPAPER · V1.0 · PHASE 1 · LAUNCHING 2026-09-12

Brics Finance

A payout token with a vault under it. Holders are paid every six hours in the tokenized assets they choose; a fixed share of every dollar buys assets that are never sold, so the payout does not fall to zero when trading does. Every payout lays a floor on the holder’s tower.

bricsfinance.fun · hello@bricsfinance.fun · MIT-licensed code

CONTENTS

1 · Abstract
2 · The payout problem
3 · Design in one page
4 · Holding and weight
5 · Revenue and the router
6 · The Foundation
7 · Settlement, end to end
8 · Proofs and the verifier
9 · Sets and the catalogue
10 · Towers and the City
11 · The Yard
12 · Threat model
13 · The trust boundary
14 · $BRICS
15 · Prior art: Mosaic
16 · Roadmap
17 · References

1 Abstract

Brics Finance is one token, $BRICS, and one loop. The creator fees its trading earns are split by a published policy: 70% is paid to holders every six hours (*Payday*), converted into the tokenized assets each holder chose (*their Set*) and pushed to their wallet; 15% buys a tokenized US Treasury note and the two broadest US equity indices into *the Foundation*, a vault that never sells principal and spends only its yield, on Payday.

Weight is a holder's balance integrated over time, to the second, from finalized chain history. Each epoch commits to a Merkle root over every holder's weight and payout; the leaf list and per-account weights are public, and an open-source verifier re-derives them from any Solana RPC. Every payday a holder receives lays a floor on their tower in the City, a public record of time held that cannot be bought.

This paper describes the mechanism as built, with the arithmetic, and states plainly what an operator can still do and what no one can promise.

2 The payout problem

A token that pays its holders from its own trading fees has one input: people trading it. That input is largest in the first days and smallest when attention has moved on. The payout therefore peaks when it is least needed to hold a community together and collapses when it is most needed. There is nothing underneath it: no asset the protocol owns, no income that does not depend on the next trade.

Two further problems follow from how such tokens are usually built:

- **Sampling.** Weighting holders by a few snapshots a day invites timing: buy before a sample, sell after. Any design that does not integrate continuously over time has to be defended against this, and usually is not.
- **Invisibility.** A holder's commitment is a number in a table. There is no durable, public record of having held through the quiet weeks, so there is nothing to lose by leaving and nothing to show for staying.

Brics Finance addresses each: it banks part of every dollar into assets that pay on their own (§6); it weights by the time integral of balance, to the second (§4); and it turns time held into something visible and permanent (§10).

3 Design in one page

LAYER	WHAT IT IS	RULE THE SYSTEM ENFORCES
The Foundation	15% of revenue buys USDY, SPYx and QQQx.	Never sells principal. Only yield above each lot's cost index is harvested, 100% to Payday. Enforced in code and by a database trigger.
Payday	70% of revenue + Foundation yield, paid every 6 hours in each holder's Set.	100% pro-rata by weight. Σ obligations = pot exactly; the database refuses any epoch that would owe more.
The surface	Towers, the City, Sets, the Yard.	Floors are append-only and laid only for deliveries confirmed on chain.

Figure 1. Three layers. The remaining 15% of revenue is buyback and burn, treasury and operations, 5% each.

One epoch

1. **Accrue.** Creator-fee claims into the fee wallet are booked as revenue at their block time and split by the router.
2. **Close.** At the epoch boundary the pot is fixed: the Payday share of this epoch's revenue plus any carry.
3. **Weigh.** Every eligible token account's time-weighted balance is computed from indexed history and summed per holder.

4. **Reconcile.** A sample of accounts is re-derived through an independent RPC provider. Disagreement halts settlement.
5. **Commit.** Largest-remainder allocation, Merkle root, manifest hash and obligations are written in one transaction.
6. **Buy and push.** Per-asset purchases are quoted and journaled, then executed; fills are split back to holders and pushed.
7. **Build.** When a holder's deliveries for the epoch confirm, a floor is laid on their tower.

4 Holding and weight

A holder is any key-controlled wallet with a $BRICS balance. Nothing is staked, locked or deposited. Token accounts owned by a program derived address (the bonding curve, liquidity pool vaults, bridge custody, multisig vaults) are excluded automatically, because a PDA is off the ed25519 curve and cannot be controlled by a key. Additional owners can be excluded by policy and are listed on the addresses page.

The weight of an account

For an epoch $[t_0, t_1)$ of 21,600 seconds, the weight of a token account with balance $b(t)$ is

$$w = \int b(t)\,dt \quad \text{over } [t_0, t_1) \qquad \text{(raw units} \times \text{seconds)}$$

where $b(t)$ is piecewise constant, changing only at the block time of a finalized transaction that touched the account. The opening balance is the post-balance of the last such transaction at or before t_0. A holder's weight is the sum over the accounts of every wallet linked to them for that epoch, and their share is their weight over the total.

How the history is obtained

The indexer takes a finalized snapshot of every token account for the mint on each pass. For each account whose balance or owner changed, it walks that account's own signature history since the last indexed signature and records the account's post-transaction balance from each transaction's `postTokenBalances`, at its block time. The snapshot identifies *which* accounts moved; the account history establishes exactly *when*. A closed account disappears from the snapshot and its history is walked once more to record the final zero.

Why timing earns nothing

Buying a balance B for Δ seconds contributes $B \cdot \Delta$. A buyer who enters one second before close earns one second of weight; with the epoch at 21,600 seconds, that is 1/21,600 of what the same balance earns by holding throughout. The only way to a large weight is balance held over time. There is no sample to hit.

> In the integration test suite, a wallet holding 50× the balance of another but buying one second before close is allocated less than the holder who bought halfway through the window. That is by construction, not by tuning.

Linking

A holder may link up to 4 wallets per chain with one message signed by both. Links take effect from the next epoch and never rewrite the past; a wallet belongs to at most one holder at a time. Splitting balance across wallets cannot increase share, because weight is additive.

5 Revenue and the router

Where the money comes from

$BRICS launches on pump.fun. The protocol does not set its trading fee: its revenue is the creator fee pump.fun pays on every trade: **0.30%** of bonding-curve volume, and after graduation to PumpSwap a creator fee that starts at **0.95%** and steps down to **0.05%** as market cap rises, per pump.fun's published schedule (last updated 2026-05-20). Creator fees accrue to the creator and move when claimed. The revenue job books every finalized transaction that credits the fee wallet *and* invokes the pump.fun or PumpSwap program, at its block time; any other inflow is ignored. Revenue is therefore booked when claimed.

> This is the protocol's most important economic constraint. A launcher that controls its own pool can charge more. The competitor discussed in §15 charges 2% on its own markets. Every figure in this paper uses the venue's real schedule.

The router

BUCKET	SHARE	USE
Payday	70%	Paid to holders every epoch.
The Foundation	15%	Buys USDY / SPYx / QQQx; never sells principal.
Buyback and burn	5%	Buys $BRICS and burns it.
Treasury	5%	Long-term protocol capital.
Operations	5%	Gas, RPC, randomness, hosting.

Every revenue event is split in integer base units: each bucket receives the floor of its share and the rounding remainder goes to Payday, so buckets always sum to the revenue exactly and dust never accrues to the treasury. The split is operating policy, served live at `/v1/policy` with a version string stamped on every routed flow.

6 The Foundation

The Foundation's share of each settled epoch buys three assets in fixed proportion:

ASSET	MIX	YIELD SHOWS UP AS
USDY, Ondo US Dollar Yield	50%	A rising redemption price backed by short-term US Treasuries.
SPYx, S&P 500 (xStocks)	30%	A rising display multiplier: dividends are reinvested by raising it.
QQQx, Nasdaq 100 (xStocks)	20%	Same multiplier mechanism.

The harvest rule

Each purchase is a lot (u, i_0): raw units bought and the asset's index at purchase: the multiplier for xStocks, the price for USDY, a constant 1 for anything without a yield mechanism. At index i the lot's *principal floor* is

$$\text{floor}(i) = \lceil u \cdot i_0 / i \rceil \qquad \text{harvestable}(i) = \max(0, \text{held} - \text{floor}(i))$$

and the invariant is $\text{held} \cdot i \geq u \cdot i_0$ at all times: the vault always holds at least the underlying it bought. Only the excess, interest and reinvested dividends, may leave, and it leaves as Payday revenue. Rounding is always against the harvester. A 5% index rise on a lot of 1,000,000 units frees exactly 47,619 units, not 47,620; the database trigger refuses the latter.

Enforcement

1. The protocol computes harvestable units in integer fixed point (indices × 10^{12}) and throws on any request above it.
2. A harvest is journaled *before* anything is sold. A `BEFORE INSERT` trigger re-checks the invariant against the database's own lot and harvest rows and refuses any row that would touch principal.
3. Lots and harvests are append-only; no code path updates or deletes them.

What it does not mean

This is a rule about the vault's holdings. It is not a floor under the price of $BRICS, and $BRICS confers no right to redeem against, withdraw or claim any Foundation asset. The Foundation's dollar value moves with equity markets; what cannot go down is the underlying it holds. Because supply only falls, the underlying held per token only rises.

Custody

Foundation purchases arrive in the payout wallet and are moved at once to a separate Foundation wallet. The payout key signs deliveries and never holds the vault. Moving yield back out for a harvest needs the Foundation key, which can be kept offline; until it signs, the harvest waits and is shown on the status page.

Illustrative arithmetic, assumptions printed

This is an illustration of the mechanism, not a projection. Every input below is an assumption chosen to show scale; none is a forecast.

ASSUMPTION	VALUE
Average daily volume on the bonding curve	$1,000,000
Creator fee	0.30%
Daily revenue → Foundation share	$3,000 → $450
Foundation principal after 90 days	$40,500
Assumed USDY yield on its 50% (≈ Ondo's published rate)	4.6% / year
Resulting yield to Payday from USDY alone	≈ $2.55 / day

The honest reading: early on, the Foundation's yield is small beside fees. Its value is that it cannot shrink and every epoch of fees adds to it, so it is the one part of Payday that grows through a quiet market rather than disappearing with it.

7 Settlement, end to end

Close and weigh

An epoch settles once its window has closed. The pot is fixed as the Payday share of revenue booked in the epoch plus the previous epoch's carry-out. Per-account weights are computed from indexed events (§4) and summed per holder, using the wallet links in force for that epoch. Owners never linked by anyone become holders of their own.

Reconcile

Before anything is written, a deterministic sample (the 25 heaviest accounts and 25 chosen by hashing the epoch number) is re-weighed from scratch through a second, independent RPC provider, reading each account's signature history and transaction post-balances directly. Any disagreement throws, the epoch stays open, and nothing is published.

Allocate and commit

The pot is split by largest remainder: each holder receives `⌊pot·w/W⌋`, and the remaining units go one each to the largest fractional remainders, ties broken by holder id. The result sums to the pot exactly and nobody receives more than the ceiling of their exact share. Weights, per-account weights, obligations (each carrying the holder's Set for that epoch), the Merkle root and the manifest hash are written in one database transaction. A statement-level trigger refuses the whole batch if the epoch would owe more than its pot.

Buy

Each holder's amount is split across their Set by largest remainder in basis points; amounts wanting the same asset are batched into one purchase. Every purchase is quoted by Jupiter with a maximum slippage and journaled with the quote, the minimum acceptable fill and each holder's contribution, under an idempotency key `payday:<epoch>:<asset>`.

Execute and push

In `plan` mode the process stops here: everything is computed, quoted and published; nothing is signed. In `execute` mode the executor first refuses to start if the payout wallet cannot fund every planned purchase, re-quotes each purchase and refuses to fill below the journaled minimum, then signs the swap, waits for finality and records the fill. Filled units are split back to holders in proportion to their contribution and pushed as `TransferChecked` transfers (with transfer-hook accounts resolved where the mint requires them), creating recipients' token accounts at the protocol's expense. Purchases and deliveries may only move forward in status; their amounts are frozen by trigger.

Opening a token account costs about 0.002 SOL, so a very small payout can cost more to send than it is worth. A holder's payout below the delivery minimum (0.02 SOL per asset in their Set at launch) is carried, with its reason, to their next payday and released once the total clears the minimum. Carry is journaled and never lost. If no route can buy an asset, that part of the payout uses the default Set for the

epoch. A purchase still unsent after an hour is quoted again at the current price rather than left stuck, and a failed delivery is retried a bounded number of times before it is shown on the status page for the operator.

Build

When every delivery to a holder for an epoch is confirmed, a floor is laid: its material is the asset family that made up most of the delivered value. Deliveries worth less than a cent lay no floor.

8 Proofs and the verifier

Encoding

```
leaf = sha256( 0x00 ‖ epoch:u64be ‖ len:u16be ‖ holder:utf8 ‖ weight:u256be ‖
amount:u256be )
node = sha256( 0x01 ‖ min(a,b) ‖ max(a,b) )
empty epoch root = sha256( 0x02 )
```

Leaves are sorted by holder; an odd node is promoted unchanged. The leaf and node prefixes prevent a node being presented as a leaf. The encoding is pinned by a snapshot test so it cannot change silently.

What is published per epoch

- The window, pot, carry, total weight, holder count, Merkle root and manifest hash.
- The manifest: every leaf `(holder, weight, amount)`.
- Per-account weights: every eligible token account, its owner and its weight.
- For any holder, a proof path to the root.

What the verifier checks

1. The manifest rebuilds to the published root.
2. Every amount equals the holder's largest-remainder share of the pot, and the amounts sum to no more than the pot.
3. Per-account weights sum to the epoch total and to the leaves.
4. Each account's weight re-derives from its own on-chain history, to the second, through an RPC the verifier chooses.
5. Every single-wallet holder's leaf equals their wallet's weight exactly.

Which wallets a holder has linked is not published, so holders with several linked wallets are covered in aggregate by checks 3 and 4 rather than individually. This is a deliberate trade between verifiability and holders' privacy.

9 Sets and the catalogue

A Set is a map from catalogue asset to basis points, summing to exactly 10,000 across at most 12 assets. Six presets cover most choices: Index (the default), AI, Mag 7, Paycheck, Compounder and Moonshot. A Custom Set covers the rest. A Set is saved with a signed message and applies from the next epoch; if an asset in it is paused by policy, that epoch pays the default Set.

The catalogue is generated from the chain, not from marketing. For every asset, the generator reads the mint with `getMultipleAccounts`, asserts its token program and decimals, and turns its freeze authority and Token-2022 extensions into plain-language disclosures. At the time of writing, most tokenized equities in the catalogue carry a permanent delegate, a pausable configuration and a default account state; several pre-IPO tokens charge a transfer fee, so deliveries arrive net of it. These powers belong to the issuers. The default Set contains no memecoins.

10 Towers and the City

A tower is derived entirely from confirmed history: one floor per epoch paid, material by asset family, footprint by share of supply (1× below 0.1%, 1.5× from 0.1%, 2× from 1%). Height is therefore a function of time alone. A holder who arrives with a large balance gets a wide building, not a tall one; a holder who stayed through quiet epochs gets a tall one. Floors are append only. A trigger refuses any floor at or below a tower's top floor, so selling stops a tower growing and never shrinks it.

The City places every tower deterministically: the Foundation at the origin, one district per Set on four avenues running out from it, and within a district the earliest holders nearest the centre. Geometry is a single isometric projection shared by the API, the site and the brand kit, rendered to SVG so the same input always produces the same drawing. Every tower has a public page and a share card.

11 The Yard

The Yard is a registry for apps whose protocol revenue flows into Payday. A registered app has its own program and vaults and can never touch the Payday pot; it sends its protocol share to a registered fee wallet, whose inflows are booked as `yard` revenue. The builder keeps a registered share of up to 50%; the rest is routed like any other revenue. Apps may read a holder's tier from their last settled share, the same checkpoint that pays Payday, to offer better terms to larger holders. The registry and SDK ship at launch; the first external app does not.

12 Threat model

THREAT	EFFECT ON HOLDERS' MONEY	MITIGATION
Payout key compromised	At most the SOL and assets in the payout wallet.	The wallet holds only the next epoch's working balance; the key lives only in the host's secret store; rotation procedure in the runbook. The key cannot touch holders' $BRICS.
RPC provider returns wrong data	Wrong weights would misallocate one epoch.	Sampled re-derivation through an independent provider halts settlement on any disagreement; the verifier lets anyone check with a provider of their choice.
Price or route manipulation during purchase	A worse fill for one asset in one epoch.	Journaled minimum fill from the quote; re-quote at execution and refuse below it; slippage cap.
Issuer freezes, pauses or seizes a catalogue asset	Deliveries of that asset delayed or blocked; delivered tokens can be moved by a permanent delegate.	Per-asset disclosures; paused assets fall back to the default Set; the default Set is index funds only.
Venue changes its fee schedule	Revenue per unit of volume changes.	Revenue is whatever arrives; nothing is promised; the site quotes the schedule with its date.
Indexer misses a transaction	An account's weight understated.	Every pass re-snapshots all accounts; any balance the history does not explain is re-walked. The affected holder can detect it with the verifier.
Database tampering	Attempted over-payment or principal sale.	Triggers refuse overdrafts, principal harvests, floor rewrites and backward epochs; published roots make any later rewrite of history evident.
Sybil splitting across wallets	None.	Weight is additive; splitting cannot increase share.
Phishing: "sign to claim"	Loss of funds to a scammer.	Payday never requires a signature; every message Brics issues says it is not a transaction.

Report a vulnerability to hello@bricsfinance.fun. See `/.well-known/security.txt`.

13 The trust boundary

Brics Finance is not trustless, and this paper does not claim otherwise.

Enforced by the chain

- $BRICS supply, transfers and balances; the mint and freeze authorities (revoked at launch on pump.fun).
- Every delivery, purchase and burn is an ordinary transaction anyone can inspect.

Enforced by the database, and checkable by anyone

- No epoch owes more than its pot; no harvest touches principal; floors only on top; history append-only.
- Every epoch's root, leaves and account weights are published and re-derivable.

Trusted to the operator

- Running the indexer, settlement and execution honestly and on time.
- Holding the payout key safely, and funding the payout wallet from claimed fees.
- Operating policy (the routing split, the catalogue, the Foundation mix, the epoch length), which is published and versioned, and changes are published before they apply.

14 $BRICS

Status	Planned — launching on pump.fun on 2026-09-12. Mint address: not deployed yet.
Chain / standard	Solana (pump.fun creates on Solana)
Supply	1,000,000,000, fixed; only decreases through buyback-and-burn
Allocation	100% to the public bonding curve. No team allocation, no investors, no presale.
Launch	pump.fun bonding curve, SOL-quoted, 2026-09-12; graduation to PumpSwap
Utility	Entitlement to Payday by time-weighted balance; a tower; a Yard tier. The site, docs, ledger and verifier are open to everyone.

What we will never claim

- A price target, an APY, a return, or that any payout is guaranteed.
- That $BRICS is an investment, a security, a fund, or a claim on the Foundation.
- That “the Foundation only grows” refers to the price of $BRICS. It refers to the underlying the vault holds.
- Any listing, partnership, audit or market figure that does not exist.

“Dividend” is used colloquially. Payday is a protocol distribution, not a corporate dividend.

15 Prior art: Mosaic Finance

Mosaic Finance ($MOS, mosaicfi.xyz) shipped first, and the idea of paying token holders in tokenized equities they choose is theirs. Their documentation is clear and their draw randomness, a commit and reveal oracle against a manifest hash fixed beforehand, is sound. Everything below is taken from their public documentation, site and API as read on 2026-09-11.

	MOSAIC (FROM THEIR DOCS)	BRICS FINANCE
Revenue	2% fee on their own official pools	Venue creator fee (0.30% on the curve; tiered after graduation)
Share to holders	80% to the draw pot	70% to Payday + 100% of Foundation yield
Asset reserve	A strategic $MOS reserve (35M floor / 75M cap)	15% into USDY / SPYx / QQQx, never sold; yield to holders
Distribution	60% proportional, 40% to a jackpot and five runners-up	100% proportional; any lottery would be opt-in
Weighting	Time-weighted; their public API shows six sampled checkpoints per 24-hour cycle	Continuous, per second, from every transaction
Cadence	1 to 3 day cycles (24h observed)	6-hour epochs
Default payout	SPYx, OpenAI, Anthropic and a memecoin, 25% each	Index Set: SPYx 60%, QQQx 40%; no memecoins
App layer	Tiles: one live game, three unopened	The Yard: an open registry with a builder share
Trust	“[It does] not make the system trustless.” (their risks page)	Same honest position, plus sampled cross-provider reconciliation and a public verifier

The structural difference is the Foundation: a protocol that banks part of every dollar into assets that pay on their own accumulates something a protocol that pays out everything cannot catch up to later. The economic disadvantage is equally structural: launching on pump.fun means a creator fee set by the venue rather than a 2% fee set by the protocol.

16 Roadmap

PHASE	STATUS	CONTENTS
1	**Launching 2026-09-12**	Payday, Sets, the Foundation, towers and the City, proofs and the verifier, statements, the Yard registry and SDK, Robinhood Chain wallet sign-in and linking.
2	Planned	A $BRICS/USDG market on Robinhood Chain behind a 2% fee hook and a lock-and-mint bridge; Robinhood Chain stock-token payouts; the first outside Yard app; an opt-in jackpot funded only by those who opt in.
3	Exploring	Holder governance over the routing split.

Nothing in Phase 2 or 3 exists today, and nothing on the site describes it in the present tense.

17 References

1. pump.fun, “Pump.fun fees”, pump.fun/docs/fees, updated 2026-05-20.
2. Mosaic Finance documentation, docs.mosaicfi.xyz: overview, money, holding, the draw, delivery, supply, multichain, tiles, risks; read 2026-09-11.
3. Mosaic Finance public API, mosaicfi.xyz/api/checkpoints and /api/rewards-view; read 2026-09-11.
4. Ondo Finance, USDY, ondo.finance/usdy; Solana mint A1KLoBrKBde8Ty9qtNQUtq3C2ortoC3u7twggz7sEto6.
5. Robinhood Chain documentation, docs.robinhood.com/chain; chain id 4663.
6. Pons launchpad, ponsfamily.com/launchpad/create; read 2026-09-11.
7. Brics Finance source: api/migrations/001_init.sql (invariant triggers), packages/core (weighting, allocation, Merkle, Foundation), sdk/verify.