

FORGE

The token that powers token creation.

Programmable token launches on Solana.
Deployment infrastructure. Transparent economics. Verifiable execution.

WHITEPAPER • VERSION 1.0
OCTOBER 6, 2026

Build the asset. Define the rules. Show the execution.

Forge is designed as a token launch and economic configuration platform on Solana. Creators use \$FORGE to deploy supported token configurations, attach economic modules, and publish verifiable launch records. The platform combines token creation, fee routing, vesting, and execution reporting in a single workflow. Its purpose is to make a token's economic behavior as visible as its supply.

Development status. This whitepaper specifies the planned Forge platform. Deployment, independent security review, and public token distribution have not been verified. Token supply, allocations, pricing, and administrative policies are preliminary design parameters pending founder approval and implementation validation. Features described here are planned unless explicitly identified as deployed; no mainnet address is published in this release.

This document communicates the platform design and development roadmap. It does not guarantee token value, investment returns, product availability, or a launch date.

01 / Executive overview

Make token economics understandable.

A token launch involves more than a name and a supply. Creators must decide how tokens are distributed, how liquidity is established, who controls administrative permissions, and where any collected fees go. When those choices are scattered across interfaces and wallets, participants struggle to understand what they are buying.

Forge's design unifies these choices in a configuration workflow backed by an on-chain registry. Creators can select supported modules, preview their behavior, authorize deployment, and direct users to a public record of the resulting configuration. The target users are token creators, launch service providers, developers, and participants seeking a clearer view of launch rules.

Deployment lifecycle

- 1. Compose.** Select token metadata, fixed supply, allocation addresses, launch venue, and compatible modules. Unsupported combinations are rejected rather than silently approximated.
- 2. Preview.** Inspect permissions, vesting, fee recipients, expected costs, and scenario examples. Simulations describe assumptions and do not predict market prices.
- 3. Quote.** Receive a short-lived deployment quote denominated in \$FORGE, with SOL network costs and liquidity requirements shown separately.
- 4. Authorize.** Review wallet-signed transactions. Forge does not request private keys. If deployment spans multiple transactions, the interface shows completed steps and recovery options.
- 5. Verify.** View the registered mint, configuration version, relevant program IDs, allocation records, and module execution history.

Initial product scope

The initial release targets standard token creation, transparent allocation records, creator vesting, and fee routing on a supported venue. Advanced modules follow after integration testing and security review. Forge's deployment registry is intended to record configuration facts; inclusion is not an endorsement of a token or its creator.

The platform's adoption thesis is straightforward: creators pay for a reliable way to deploy understandable token economics, while participants gain a consistent way to inspect them. Product performance is measured by completed deployments, repeat creator usage, and accurate execution accounting.

02 / Economic modules

Modules define what supported, collectible launch revenue does. A module cannot capture fees from unrelated trading venues merely because the token was created through Forge. Every configuration must identify the fee source, collection authority, and destination addresses.

Module	Defined behavior	Release scope
Creator Vesting	Releases a designated allocation over a published schedule.	Initial release
Treasury Router	Routes collectible fees among declared recipients according to fixed shares.	Initial release
Liquidity Builder	Adds eligible collected fees to a compatible pool under explicit ownership rules.	Later, after integration testing
Scheduled Buyback	Uses a capped reserve for time-spaced purchases with slippage limits.	Later, after security review
Buyback and Burn	Purchases a supported token and burns received units; publishes receipts.	Later, after security review

Configuration and execution policy

Each module records an allocation rule, execution threshold, supported assets, authorized actions, and failure behavior. Splits must sum to 100%. Minimum execution thresholds limit disproportionate transaction overhead.

Launch policies are designed to be fixed by default. If a configuration permits changes, the interface must show who can make them, any delay, and the affected parameters. A fixed configuration does not by itself make an upgradeable program immutable; both policy controls and program upgrade authority must be disclosed.

Planned buyback modules enforce maximum spend per interval, a minimum reserve, quote freshness, and slippage limits. Failed execution must leave funds in the vault for a later attempt; it must not redirect them to an undisclosed recipient.

Liquidity additions do not create a price floor. Token burns reduce supply but do not guarantee demand or appreciation. These modules provide allocation and execution functions; they do not provide insurance or guaranteed yield.

03 / The \$FORGE token

A token consumed by a creation workflow.

The primary planned utility is payment for Forge deployments. A creator obtains \$FORGE, authorizes the quoted deployment fee, and receives a registered launch. The fee is split between permanent burns and the operating treasury. This gives the token a defined use without requiring trading taxes on the \$FORGE token itself.

Utility	Planned utility
Deployment payment	Every Forge mainnet deployment requires a quoted \$FORGE service fee. SOL gas and liquidity funding are separate.
Advanced access	An optional time-locked balance unlocks batch tools, saved templates, and API limits. Locked balances earn no promised return.
Governance	A later release may use time-weighted locked balances for bounded votes over future platform parameters. Existing fixed launch rules remain unchanged.

Deployment pricing and fee allocation

The preliminary service-price schedule is \$25 equivalent for a basic deployment and \$100 equivalent for an advanced deployment, payable in \$FORGE. Final pricing is subject to validation. Quotes specify the token amount, configuration hash, payer, expiration, and supported deployment actions. SOL network fees and creator-supplied liquidity are separate from this service charge.

For a fee of F tokens, the planned split is $0.50 \times F$ burned and $0.50 \times F$ sent to the operating treasury. If a valid quote is \$0.01 per \$FORGE, a \$25 deployment costs 2,500 \$FORGE: 1,250 burned and 1,250 retained. This example is arithmetic, not a price forecast.

Pricing in dollar-equivalent terms helps keep deployment affordable if \$FORGE changes price. It also means a higher token price reduces the token quantity consumed per deployment. Deployment growth cannot be translated directly into a token-price target.

Demand and supply dynamics

Demand depends on creators choosing Forge over alternatives. Creators can buy just enough tokens for a deployment, and treasury spending can return tokens to circulation. Optional locks reduce immediately available supply temporarily; they do not eliminate future selling. The product must earn demand through usefulness, reliability, and distribution.

Paid deployment requires a manipulation-resistant pricing source and sufficient liquidity. Invalid or stale quotes are rejected. Service-fee settlement must prevent duplicate charges and reconcile incomplete deployments; the implementation specification will define the charging boundary before release.

04 / Token economics

The preliminary token model uses ticker **\$FORGE** and a maximum initial supply of **1,000,000,000 tokens**. The intended token standard is SPL, with no transfer tax and no freeze authority. The design requires mint authority revocation after the approved supply is minted. Actual supply and authority state will be disclosed and verifiable before public distribution.

Allocation	Share / tokens	Planned release policy
Public distribution	45% / 450M	Method and initial float remain undecided; publish all terms before launch.
Ecosystem incentives	20% / 200M	Up to 48-month budget; grants tied to genuine usage and documented deliverables.
Liquidity allocation	15% / 150M	For disclosed pools; quote-asset funding, LP ownership, and any locks must be published.
Operating treasury	10% / 100M	24-month linear vesting; expenses reported quarterly.
Team and contributors	10% / 100M	12-month cliff, then 24-month linear vesting. No undisclosed side allocations.

The allocation totals 100%. The preliminary allocation includes no separate private-sale bucket. The distribution method, launch price, circulating supply at launch, market-making arrangements, and wallet addresses are unresolved. Therefore, this paper cannot calculate an initial market capitalization or fully diluted valuation.

Release and treasury discipline

Under the planned schedule, team tokens remain locked during the cliff, then vest evenly over the following 24 months. Treasury and incentive releases will be documented on a public calendar. Unused incentive tokens remain in the disclosed allocation; a scheduled budget is not a requirement to distribute it.

Initial liquidity requires both \$FORGE and a quote asset. Reserving 150 million tokens does not itself fund a liquid market. The source of SOL or stablecoin funding and the person controlling liquidity withdrawals must be established before launch.

The operating treasury receives vested allocations and its share of deployment fees. Those funds pay for engineering, security, infrastructure, and support. Treasury receipts are not holder dividends. The described utility does not grant token holders equity, a redemption right, or an entitlement to operating revenue.

The incentive program targets useful templates, verified bug reports, and completed development work. Eligibility will exclude artificial usage intended solely to farm rewards, with criteria published before any program begins.

05 / Architecture and security

A modular deployment architecture.

The architecture defines six components: a web composer; a deployment coordinator; a launch registry; per-launch vaults; approved module programs; and an indexer that presents on-chain history. An execution service can submit eligible module transactions, but funds and permissions remain constrained by program rules.

The coordinator validates a configuration and records deployment state. A launch vault must only move funds through actions explicitly permitted by its module. The registry binds each launch to its configuration hash, module version, mint address, and control authorities. The indexer is a convenience layer; users must be able to verify relevant facts independently.

Integration strategy

Meteora Dynamic Bonding Curve is a candidate integration, not a confirmed partnership. Its official documentation describes configurable curves, launch fees, and migration into DAMM liquidity [1]. The planned integration uses a validated subset of these capabilities; the underlying venue retains its own rules, fees, and dependencies.

Solana Token-2022 transfer hooks can invoke custom logic during token transfers [2]. Integration requires clients to supply the hook accounts; missing accounts can cause transfers to fail [3]. The initial release does not depend on transfer hooks. Any later hook-enabled configuration requires a published compatibility matrix. Venue fees, transfer fees, and Forge deployment fees must be explained separately.

Required controls before public deployment

Authority management. Publish upgrade, mint, freeze, vault, and treasury authorities. Use a disclosed multisig for administrative powers and a planned 72-hour delay for ordinary program upgrades. Emergency powers, if any, must be narrowly specified and visibly reported.

Validation. Test allocation conservation, vesting boundaries, authorization, failed execution recovery, quote expiration, slippage limits, and repeated calls. Simulate user transactions before requesting signatures.

Independent review. Obtain an external security review covering custom programs and integration assumptions. Publish scope, findings, remediation, and deployed program version. An audit is not a guarantee of safety.

Operational resilience. Plan for RPC outages, stale indexer data, executor downtime, and venue migration failures. Display pending versus confirmed states, avoid success claims before confirmation, and bound value exposed during beta.

06 / Governance and roadmap

Development milestones.

Phase	Deliverable	Exit condition
1. Validate	Creator interviews, pricing tests, configuration prototype.	Evidence that independent creators want the workflow and will pay for it.
2. Demonstrate	Devnet composer, registry, vesting, and fee routing.	Repeatable end-to-end deployments; documented failure recovery.
3. Review	Published source, tests, authority map, external security review.	Material findings resolved; deployment limits and incident procedures defined.
4. Release	Limited mainnet beta with basic paid deployments.	Independent usage, reliable receipts, reconciled fee accounting.
5. Expand	Advanced modules, API, template marketplace, governance.	Each addition passes integration and security review.

Public distribution is contingent on a working paid-deployment path, documented token economics, security review, and deployment readiness. The roadmap uses acceptance milestones rather than fixed dates. Mainnet program addresses and the official mint address will be published through verified project channels when available.

Bounded governance

The initial administration model is a disclosed multisig. Token governance is a later milestone, with scope limited to future template approvals, incentive budgets, and bounded deployment parameters. Before activation, a separate governance specification will define vote accounting, quorum, delegation, proposal eligibility, and execution delays.

The intended governance scope does not permit holders to rewrite existing fixed launch rules, seize creator assets, or spend isolated launch vaults. Any upgradeable component must disclose the realistic reach of its authority; interface promises cannot override program privileges.

Public reporting

Reporting is designed to include successful paid deployments, distinct paying creators, repeat usage, gross service fees, burned tokens, treasury receipts, module failures, and operational incidents. Subsidized activity is identified separately. Forge token trading volume is not reported as platform revenue.

Forge's long-term objective is a reusable economic configuration layer for token creators. Expansion depends on demonstrated adoption of the basic deployment product, not on token trading activity. Affordability, transparent permissions, and verifiable execution remain the core design principles.

07 / Disclosures and references

Material risks

Adoption and liquidity. Creators may prefer competing products, or token-based payment may add friction. \$FORGE may be illiquid, highly volatile, or lose all market value. Neither a fixed supply nor burning provides a floor.

Programs and dependencies. Bugs, compromised keys, malicious upgrades, unsafe configurations, oracle manipulation, or external venue changes can result in lost funds. Execution services and RPC providers introduce availability dependencies.

Distribution and control. Concentrated ownership, vesting releases, treasury selling, and thin initial liquidity can materially affect the market. A configuration registry improves visibility but cannot certify that a creator is honest or that a launch will succeed.

Public-launch obligations. The operator must obtain jurisdiction-specific advice on token distribution, product fees, marketing, taxes, and any applicable restrictions before release. This document makes no determination about the token's legal classification.

Launch disclosures

Before public distribution, Forge's launch disclosures must identify the operator and contributors, development funding, distribution method, initial circulating supply, liquidity funding and control, final fees, pricing source, administrative authorities, review results, and verified mint, program, treasury, and vesting addresses. None is established solely by this whitepaper.

Version updates will record material changes to the design parameters and development status. Descriptions of planned features must not be interpreted as evidence of deployed functionality. Final on-chain parameters and published authority disclosures take precedence over preliminary numerical examples in this release.

Primary technical references

[1] Meteora official documentation repository: "What is DBC?"
github.com/MeteoraAg/docs — [Dynamic Bonding Curve overview](#)

[2] Solana documentation: "How to use the Transfer Hook extension."
solana.com/docs/tokens/extensions/transfer-hook

[3] Solana Foundation documentation source: transfer-hook integration.
github.com/solana-foundation/solana-com — [integration guide](#)

References reviewed October 6, 2026. They establish available technical building blocks, not a Forge integration, endorsement, or completed implementation.