StarloOS / STARLO

Technical, product and token-economy whitepaper

VERSION 1.0 / 22 OCTOBER 2026
HYPERDRIVE SOFTWARE 0.8.1
WINDOWS ARTIFACT 0.8.0 - UNSIGNED CANDIDATE

TOKEN NOT LIVE
CONTRACT ADDRESS NOT PUBLISHED

A detailed account of the architecture, execution controls, proposed service utility, economic model, risks and launch requirements.

Pre-launch design and disclosure draft. Not a completed regulatory offering document, investment-return promise, security certification or proof of live payments. No verified official trading pair or purchase route is supplied.

34 chapters / 7 appendices / source-linked disclosures

Read this first

Read this first. This paper describes an existing software alpha and a proposed token-funded service. It is not proof of a deployed token, audited contracts, operating liquidity, regulatory approval, or commercial traction. It does not establish an offer to sell, ownership in an issuer, a claim on revenue, guaranteed access to future services, or an entitlement to investment returns. No official purchase destination or contract address is supplied in this edition. Do not send funds to an address merely using the STARLO name. This technical paper is not a completed regulatory offering document. An identified responsible issuer, jurisdiction-specific review, completed disclosures and any required filing remain necessary before a public offer or admission to trading.

The body of this document distinguishes implemented software from configuration-dependent capabilities, proposed designs and unverified launch conditions. Figures in economic illustrations are assumptions, not prices, traction or return forecasts. The source and document manifests are published with the web edition.

Publication and provenance

This edition covers the supplied Hyperdrive 0.8.1 source baseline and its pinned Windows 0.8.0 candidate. The documentation update does not alter the agent engine, activate token payments, replace the native artifact or certify the live website. The source references identify the underlying files; the final appendices provide review and launch criteria.

Contents

01. Executive summary 5
02. How to read this paper 5
03. Problem, audience and product thesis 6
04. Product scope and current-state register 6
05. System design principles 7
06. Architecture and trust boundaries 8
07. The agent execution lifecycle 9
08. Specialists, workflows and available tools 9
09. Agent Academy: teaching, practice and evaluation 10
10. Knowledge, memory, files and provenance 11
11. Crypto Research Deck and read-only chain data 12
12. Deployment, distribution and availability 12
13. Service credits and execution accounting 13
14. STARLO identity, supply draft and holder rights 14
15. Token utility and the reason to hold 15
16. Draft smart contracts and their limits 15
17. Proposed payment-to-credit protocol 16
18. Settlement, exceptions and reconciliation 16
19. Service pricing and economic sustainability 17
20. Token demand, liquidity and valuation boundaries 18
21. Allocation, vesting and treasury policy 19
22. Governance, administration and accountability 20

23. Optional creator and service economy 20

24. Security model and operational controls 21

25. Privacy, data lifecycle and user control 21

26. Performance, reliability and scaling boundaries 22

27. Evidence, evaluation and transparent reporting 22

28. Risk register 23

29. Legal, issuer and disclosure readiness 24

30. Environmental and resource considerations 25

31. Delivery roadmap and release gates 25

32. Public launch status, navigation and anti-impersonation 26

33. Open decisions and accountability register 26

34. Frequently asked questions 27

Appendix A. Capability and dependency checklist 29

Appendix B. Mission and payment invariants 29

Appendix C. Illustrative end-to-end research mission 30

Appendix D. Contract, ledger and failure test plan 30

Appendix E. Glossary 31

Appendix F. Source register and provenance 32

Appendix G. Document control 33

01. Executive summary

StarloOS is a visual workspace for configuring AI specialists, assigning bounded jobs, inspecting requested tool actions, and retrieving the resulting files. Its central product principle is straightforward: the user should be able to understand what an agent attempted, which permission allowed it, what it cost in the application's accounting, and which parts of the result still need judgment. The interface is a control surface for software execution, not evidence that intelligence or work exists behind an animation.

The delivered Hyperdrive source includes an agent execution loop, specialist delegation, reusable ordered workflows, knowledge notes, approved memory, bounded file tools, a public web reader, optional integrations, a read-only chain snapshot tool, credit reservations and human outcome review. There is a local runtime and a separately configurable hosted backend. The Windows artifact is an unsigned installation candidate, not a signed, independently verified consumer release. These statements describe source capabilities, not a guarantee of successful operation with every provider, website, computer or hosted account. [C1-C8]

STARLO is the proposed service token. Its intended initial use is purchasing credits for managed AI execution. A customer would pay a verified token quote, receive an internal service balance and spend that balance on permitted work. Token ownership, service credits and model usage units are different things. Purchasing a token would not itself train an agent or provide an unlimited claim on compute. The payment bridge is a proposed design; this edition does not enable it. [C9-C11]

The strongest product hypothesis is a research crew for people following crypto projects: collect approved public evidence, compare it with previous research, identify unanswered questions and produce files the user can inspect. This is not an autonomous trading service. The current read-only chain tool neither signs transactions nor controls a wallet, and it does not establish project identity, asset value or contract safety. [C5]

The project must earn adoption through accepted useful outputs, repeat use, transparent cost and reliable operation. It does not have a demonstrated new foundation model, a decentralized compute network or a proven economic advantage simply because it uses agents and a token. No verified paying-user count, revenue series, retention cohort, token distribution or exchange liquidity is established by this document. A roadmap is a set of proposed acceptance gates, not a claim that those gates have been passed.

02. How to read this paper

Four labels distinguish materially different kinds of claims. **Implemented** means a behavior is present in the reviewed source. **Configuration-dependent** means the code requires external credentials, services, permissions or infrastructure. **Proposed** means a design for later implementation or an operator policy awaiting adoption. **Unverified** means this edition lacks the evidence needed to assert successful operation or a factual launch condition. An implemented capability can also be configuration-dependent and unverified against a live service.

References beginning with **C** identify bundled source files or release records. References beginning with **R** identify external primary documentation. The appendix records the software archive hash and selected file hashes so readers can distinguish this paper's baseline from subsequent releases. A content hash identifies bytes; it does not establish that those bytes are safe, correct or commercially deployed.

This edition is intentionally explicit about unresolved matters. There is no identified issuer entity, registered address, responsible management statement, approved allocation table, chosen token deployment network,

verified CA, funded vesting schedule, exchange pair, liquidity plan, opening price or public-sale timetable in the supplied release. These are not editorial blanks to fill with optimistic assumptions. They are decisions and disclosures required before any relevant launch.

A later edition must explain material changes and keep a dated archive. Changing a website label cannot retroactively change a customer's purchased service terms or the actual behavior of a deployed smart contract. Where the site, paper, source and chain disagree, the inconsistency must be resolved rather than presented as a harmless branding difference.

03. Problem, audience and product thesis

An AI chat response is not the same thing as a completed job. Useful work often requires acquiring evidence, performing calculations, producing structured outputs and reviewing the result. Users also need to know when the model refused, guessed, lost context, encountered an inaccessible page or exhausted a budget. StarloOS aims to make these boundaries visible while offering a reusable environment for recurring work.

The initial audience hypothesis is a technically curious crypto user or small research team that monitors public projects and wants sourced briefs rather than unexplained predictions. Additional potential users include creators preparing documentation, operators organizing policies, and small teams assembling decision materials. These are proposed market segments, not verified customers.

A suitable first task is: "Review these selected official sources, compare the findings with my previous brief, write a source table, and separate confirmed changes from questions requiring follow-up." A poor initial task is: "Trade my wallet until it produces a return." The former has inspectable inputs and deliverables. The latter combines model uncertainty, market risk and irreversible financial actions that the delivered product does not support.

The product competes on coordination, control, evidence and usability. A defensible advantage would have to develop from reliable workflows, accumulated evaluation evidence, maintained integrations, useful distribution and customer trust. The interface's science-fiction visual language is a design choice, not a technical moat. The mere existence of four named agents or a fungible token is not a novel algorithm.

Product-market fit remains a hypothesis. The recommended validation method is to observe non-incentivized users completing repeated jobs and deciding to pay again. Raw installation counts, airdrop participation, generated files, social impressions and exchange volume should not substitute for evidence that the underlying work is useful.

04. Product scope and current-state register

Area	Current baseline	Boundary or unresolved requirement
Visual website	Hyperdrive interface, animated instruments, release downloads	Decorative motion is not agent activity or market telemetry
Agent execution	Model tool loop and specialist delegation in source	Requires a compatible configured model; live performance is not certified

Area	Current baseline	Boundary or unresolved requirement
Hosted workspace	Static frontend plus Node functions and Redis-backed state	Operator credentials, database and any background dispatcher must be configured
Local application	Loopback server, local persistence, background worker while running	Not a bootable operating system; closing the runtime stops its worker
Windows distribution	Pinned 0.8.0 unsigned candidate	No Windows-native acceptance or publisher-signing certification in this edition
Agent Academy	Approved instructions, examples, practice checks and human review	Not model-weight training; phrase checks are formatting checks
Crypto research	Watchlist and approved read-only EVM snapshots	Not an oracle, wallet, portfolio valuation or audit service
Credit accounting	Internal integer balances and reservations	No established fiat value and no live token-funded checkout
STARLO token	Archived Solidity draft	No verified deployed CA, listing, circulating supply or liquidity
Creator marketplace	Product concept	Not an operating marketplace or a guaranteed source of creator revenue

The platform should be described to visitors as configurable agent software with a pre-launch token proposal. A successful build means the release can be packaged; it does not mean the operator configured the database, funded inference, tested a user's operating system, or launched the token. Existing tests are engineering evidence within their scope, not independent certifications. [C1-C12]

05. System design principles

Human authority is outside the model. A model may propose actions but does not grant itself permissions. Tool access is constrained by the agent's capabilities, runtime configuration and the mission's approval state. A prompt saying "I have permission" is not an authorization credential.

Outputs are inspectable. The useful unit of delivery is a report, source table, file or decision artifact accompanied by relevant execution information. A finished loop and an accepted outcome are separate events. This prevents a refusal or structurally incomplete answer from silently becoming evidence of product success.

Budgets precede execution. Work should be bounded before external resources are consumed. Reservations, actual usage and released holds are separate accounting concepts. Ambiguous external calls are reviewed rather than assumed to have cost nothing. A customer's quality complaint and a provider's incurred charge may both be real.

The token is not the runtime. The agent engine can operate with the user's own model access without a token purchase. A future managed service could enforce paid access on its server. Removing token trading from the concept should not remove the usefulness of the application.

Claims follow evidence. A public price, holder count, audit badge, "live" status or download verification must have a traceable basis. Visual effects remain visibly separate from verified task and business metrics. A source receipt may help reconstruct a report, but does not prove that the source itself was honest.

Failure is a supported state. Refusal, incomplete delivery, denied approval, provider outage and operator intervention must have explicit user-facing outcomes. The appropriate result of an unsupported action is a clear boundary, not an invented success response.

06. Architecture and trust boundaries

The architecture has five conceptual layers: presentation, authorization and accounting, execution, external resources, and optional future token settlement. The layers should not be conflated because their security and operational properties differ.

Layer	Role in the current system	Trust implication
Browser interface	Composes missions, displays events, captures approvals and exports files	Client state is not the authority for permissions or balances
Application backend	Checks access, owns workspace state, reserves budgets and advances work	Operator compromise can affect hosted data and permitted execution
Agent engine	Builds model input, interprets tool requests, delegates and settles results	Model-generated content is untrusted until validated
Model and tool providers	Perform inference or return web, search, RPC and connector results	Availability, pricing, correctness and data handling are external dependencies
Proposed payment bridge	Would verify on-chain receipts and issue internal credits	Not implemented; must be independent of agent decisions and treasury keys

The browser sends authenticated requests to the backend. The backend snapshots the selected task, agents, tools, knowledge and budget. An advance request claims one execution action. The external call occurs outside optimistic state-update retries, and a matching claim identifier is required to settle its result. This separates a provider action from the logic that may retry a state update. [C1-C3]

Local execution uses a loopback-only server and file-backed storage. The hosted path uses Vercel functions and a Redis REST adapter with compare-and-set updates. These are operator-controlled services, not a decentralized consensus layer. The application is not an L1 or L2 blockchain, and its state is not independently reproduced by a network of validating operators.

The source limits workspace size and concurrency. It is designed for controlled pilots, not a proven high-volume multi-tenant service. Moving to normalized transactional records, object storage, durable queues and stronger tenant-boundary testing is a scaling requirement. Distributing more tokens does not create that infrastructure.

07. The agent execution lifecycle

A mission begins with an explicit brief, selected agent or workflow, permitted capabilities, optional reference material, maximum budget and any required filenames. The system records a snapshot so later edits to an agent do not silently change a mission already in progress. Permission changes should be made visible rather than interpreted as retroactive consent.

Execution alternates between model decisions and permitted tool actions. The model may return text, request a tool, delegate a subtask or ask the user a question. The runtime validates tool names and argument shapes, checks granted capabilities, requests approval where required and returns the resulting data to the next step. The application's model-adapter pattern uses tool calling; the provider produces requests, while application code executes the tools. [C1-C3; R1]

A specialist delegation creates a separate frame of conversation while sharing mission files and the remaining mission budget. Delegation is bounded. It is not an unrestricted population of agents, and the model cannot create infinite independent spending commitments. Ordered workflows select specialists in sequence; the delivered interface is not a general conditional directed-graph engine.

An approval is specific to an action and its arguments. Denying the request must not execute it. A later request with a different destination, file content or connector payload is not automatically covered by an earlier approval. Source content, a model's explanation and a community message cannot approve an action on the user's behalf.

The execution may end as completed, incomplete, refused, failed, cancelled or held for review. A completed execution still needs human acceptance. Required deliverable checks inspect declared filenames and nonempty content; they do not certify the semantic accuracy of those files. The current refusal detection includes explicit flags and a conservative text heuristic, not an omniscient refusal classifier. [C4]

Quality status and charging must remain separate. If a provider consumed resources before a refusal or failure, the record should retain that usage. A commercial operator still needs a clear refund and dispute policy; the current accounting implementation does not constitute such a policy.

08. Specialists, workflows and available tools

The default crew provides four understandable starting roles. **Scout** investigates evidence. **Prism** analyzes calculations and tradeoffs. **Quill** assembles deliverables. **Gate** reviews claims, omissions and references. These are configurable instructions and capability profiles over external models, not four proprietary foundation models. A reviewer agent can make mistakes just like a writer agent.

Tool family	Implemented behavior	Important limit
Public web	Retrieve approved public HTTPS pages and extract text	No logged-in browser, JavaScript interaction or arbitrary private-network access
Search	Optional operator-funded Brave Search discovery	Snippets are leads, not evidence that a claim has been verified

Tool family	Implemented behavior	Important limit
Files	List, read and write bounded UTF-8 files in a mission namespace	Does not grant arbitrary access to the user's disk
Calculation	Bounded arithmetic parser	Not an unrestricted code evaluator or financial data source
Memory	Search selected notes and save approved memories	Basic keyword retrieval; private content can be included in model context
Delegation	Pass a bounded subtask to a permitted specialist	Shared mission constraints still apply
Questions	Pause for a user decision or clarification	No implied approval from a website or model message
Chain snapshots	Read selected public state on configured EVM networks	No signing, sending transactions, valuation or safety score
Remote MCP	Discover and call tools on owner-configured servers	Compatibility, credentials and exact-action approvals are required
Local Python	Optional owner-only Docker execution	Requires explicit local setup; not available as arbitrary Vercel shell execution

The agent can use only tools exposed by its capability set and current configuration, and execution checks remain in place even after the model proposes a call. The current schema is deliberately narrow. This reduces accidental scope but cannot eliminate prompt injection, malicious tool behavior or poor human approval decisions. [C3]

A useful workflow is a reproducible recipe: clear inputs, allowed resources, expected outputs and review criteria. The proposed future library should emphasize tested outcomes and versioned changes rather than the number of templates available. A template's popularity is not proof of security or factual reliability.

09. Agent Academy: teaching, practice and evaluation

The Academy is a local teaching interface. A user can supply instructions, approved example inputs and outputs, and practice tasks. In the current source, teaching has bounded lengths, up to five approved examples and up to five practice tasks, with revision checks to avoid overwriting a concurrent edit. Teaching text explicitly does not create extra tool permissions. [C6]

This is personalization through context. It does not change model weights, create a new trained foundation model or guarantee that the agent remembers every interaction. Supervised fine-tuning is a separate provider training process using example data; that service is not integrated into the delivered Academy. [R2]

The evaluation distinction matters. A phrase such as "30-day refund" can appear in an answer that denies the existence of that policy. Literal contains/excludes checks therefore test formatting or limited output requirements, not understanding. The application separates format results from accepted quality and asks a person to review the answer before accepting it. [C6]

The proposed stronger evaluation practice is to define a rubric before testing, reserve unseen examples, record the model and prompt revision, compare outputs against evidence, include refusal and hostile-input cases, and inspect both false successes and unnecessary refusals. These are evaluation procedures to adopt, not capabilities silently performed by the current phrase checker.

Teaching material can contain confidential policies or examples. Users should minimize identifying details and should not place secrets, wallet keys, credentials or restricted personal data into examples. The external provider may receive material included in a mission. A local interface does not automatically mean local inference.

No "training certificate," accuracy percentage or qualification badge should be issued merely because a few examples passed. A public performance claim would need a named dataset, task definition, sampling method, model version, evaluation criteria, failure count and uncertainty. Benchmark improvement would still not guarantee future behavior on different tasks.

10. Knowledge, memory, files and provenance

The current knowledge layer stores bounded user-supplied text and selected documents. The memory tool searches approved notes and selected knowledge by keywords and returns a limited set. It is not a large-scale semantic document index, and a negative search result does not prove relevant information is absent. [C3]

Memory saves require approval. This lets users distinguish a fact they approved, a preference they chose and an unsupported model inference. Good memory entries should contain a clear title, context and an indication of uncertainty. The future product should offer expiry, source attribution and revision history for memories that can become stale. Those extensions are recommendations, not current guarantees.

Generated text files reside in a mission-scoped namespace. The path validator rejects traversal and unsafe path forms. This is different from granting an agent arbitrary filesystem control. Exports may contain sensitive source material or generated text; downloading a ZIP does not anonymize it. The user remains responsible for where exports are shared.

Evidence records can include a source URL, retrieval timestamp, normalized text and content hash. These help establish which extracted bytes informed a report. They do not prove the source author is trustworthy, that the source was complete, or that every model claim follows from it. Dynamic pages, inaccessible resources and text extraction can omit important context.

A proposed publication standard is to attach each material research claim to a source or explicitly mark it as inference, assumption or unanswered question. A citation identifier that exists in a source table is only a structural property. Semantic entailment still needs review, particularly for financial, legal or security claims.

11. Crypto Research Deck and read-only chain data

The Research Deck starts with a user-maintained watchlist and selected public sources. A mission can request a brief and a source table, compare with supplied earlier research, and separate changes from unknowns. Useful outputs include a dated project update, comparison of published documentation or a list of claims requiring further verification. The feature is not a price-prediction engine and must not imply that public facts establish suitability for an individual investor. [C5]

The chain snapshot tool supports operator-configured endpoints for Ethereum, Base and Base Sepolia. These supported research networks are not a selection of the eventual STARLO deployment chain. Each approved snapshot checks the endpoint's reported chain identifier, requests a finalized block, reads bounded public state and checks block consistency. It can report a native balance, code presence and limited ERC-20-shaped metadata reads. Failed metadata reads are not evidence of fraud, and successful reads do not authenticate a token.

An address supplied by a user is not automatically the official address of a project. Code presence does not establish that code is safe, immutable or free of privileged controls. A balance is not a valuation. A finalized block returned by an RPC provider is still dependent on the provider's correctness and the selected network's behavior. The tool sends no transactions and requests no wallet signatures. [C5]

The proposed research standard includes an explicit as-of timestamp, named source scope, missing-source list, comparison baseline and uncertainty section. "Nothing found" should mean the stated sources produced no finding, not that no relevant event exists anywhere. Stale information must be distinguishable from a current observation.

Autonomous trading, tax advice, contract auditing, identity verification and custody are outside this release's scope. A broader product could add specialized integrations after review, but this paper does not grant agents authority to move assets. Keeping the first crypto workflow read-only is a deliberate risk boundary, not an invitation to bypass controls through a connector.

12. Deployment, distribution and availability

The website is deployable as static assets and Node functions on Vercel. Building the source recreates public download archives and verifies them against the release metadata. The source allowlist excludes ordinary private runtime state and generated archives to avoid recursive packaging. A maliciously embedded secret in an allowed source file would still be a disclosure risk; packaging is not secret-scanning certification. [C7-C8]

A public website does not activate a hosted private workspace. The operator must configure database access, owner authentication and the selected model provider. Optional search, RPC and connectors need their own configuration. Background hosted work requires an appropriate dispatcher; a successful function response does not promise that an unbounded process remains alive.

The local application runs a loopback server, stores data locally and uses an owner worker while the process is running. It can avoid a cloud database, but its configured model may still be remote. The current Windows candidate obtains a pinned runtime during setup rather than bundling a fully offline AI stack. It does not include a pretrained model or prepaid inference. The website source is 0.8.1 while the retained Windows artifact is 0.8.0; this document does not silently upgrade or re-sign that executable. [C7-C8]

Operational acceptance requires more than unit tests: installation on a supported clean system, configuration with a funded model, a real permitted tool action, downloaded output, restart persistence, approval denial, cancellation, budget behavior and shutdown. The release records contain tests with simulated external responses and constrained browser verification. They do not establish a live hosted deployment, independent penetration test or Windows acceptance result.

The proposed production availability policy must specify maintenance, provider outages, regional restrictions, support response expectations and how held jobs are resolved. No uptime service-level agreement is promised in this edition. The platform should degrade visibly rather than display a false connected status when a provider or database is unavailable.

13. Service credits and execution accounting

Service credits are internal accounting units. They are not blockchain tokens, cash deposits, guaranteed dollar balances or proof of profit. The current application uses integer accounting with 10,000 internal units per displayed credit. This precision is an implementation choice, not an exchange rate between STARLO, dollars or model tokens. The current owner can allocate pilot credits; no live purchase mechanism establishes their commercial price. [C2]

For an execution budget B, the design distinguishes available balance, reserved balance and settled consumption. Reserving B reduces spendable balance while preserving the fact that B has not necessarily been consumed. After verified usage U, unused reserved units may be released. The simplified relationship is `released = reserved - settled`, subject to holds and the accounting invariants. No release should make settled usage disappear.

The implemented engine uses claims to avoid two callers performing the same action, and persistent grant receipts are separate from the shortened activity ledger. Permanent replay detection for administrative grants is an important repair, but it is not an implemented blockchain-payment receiver. A future payment ledger requires its own unique identifiers, canonical transaction evidence and recovery procedures. [C2]

Usage-metering policy must disclose which model inputs, outputs, tools and infrastructure are charged, how rounding works, and whether failed or cancelled calls incur costs. Provider tokens are a measurement of inference input and output. They are not the STARLO asset. A provider can change rates independently of token-market prices.

The proposed customer balance should be nontransferable and used only for the defined service. Any non-redemption, expiry, refund or cancellation policy is subject to applicable law and must be approved and disclosed before paid operation. A technical implementation must not silently turn a proposed policy into a binding customer agreement.

14. STARLO identity, supply draft and holder rights

Field	This edition's disclosure
Project and application	StarloOS
Proposed token name	Starlo
Proposed ticker	STARLO; ticker uniqueness is not established
Reference technology	ERC-20-style Solidity draft; deployment chain not selected here
Draft initial supply	100,000,000 STARLO with 18 decimal places
Draft initial recipient	One nonzero treasury address supplied to the constructor
Verified contract address	Not published; CA NOT LIVE
Circulating supply and holders	Not established
Official pair and purchase URL	Not verified; both controls remain not live
Minting beyond construction	No additional public mint method in the reviewed token source
Transfer taxes, blacklist, yield	Not implemented in the reviewed token source
Ownership or revenue claim	No company equity or automatic revenue entitlement implemented

The ERC-20 standard defines interfaces for fungible token balances, transfers and approvals. It does not establish token value, liquidity, issuer credibility or an investment right. A token's name and symbol are not globally unique identifiers; the network and verified contract address matter. [R3]

The draft's constructor mints its entire initial supply to the supplied treasury. It does not distribute allocations, fund locks, create a market or enforce a commercial tokenomics document. That concentration is a material control issue until actual transfers and restrictions are disclosed. The 100 million quantity is an existing draft parameter, not a final approved issuance policy. [C9]

There is no statement here that STARLO is a security, is not a security, qualifies for a particular exemption, or will be legal to buy in every jurisdiction. Rights arise from actual contracts, lawful terms and applicable law - not the word "utility." No revenue share, dividend, guaranteed buyback, fixed redemption or guaranteed yield is included in the reviewed design.

15. Token utility and the reason to hold

The proposed initial utility is access to managed execution purchased using STARLO. The operator would provide infrastructure and provider access, and its server would enforce the purchased credit balance. The downloadable bring-your-own-key runtime is a separate path: a user paying a model provider directly does not inherently generate STARLO demand.

Payment utility is not the same as holding utility. A person may acquire tokens only immediately before buying credits; the operator may sell received tokens to pay expenses. Those actions can create both buying and selling. Product revenue therefore cannot be equated with token-holder returns. Neither usefulness nor supply scarcity guarantees appreciation.

Potential future membership benefits could include bounded service discounts or access to maintained specialist workflows. No tier thresholds, staking contracts, reward rates or benefits are activated in this edition. Any benefit must have a funding source, cost ceiling, eligibility policy and withdrawal or revision rules. A one-time holding decision must not create an unfunded promise of unlimited ongoing inference.

A creator marketplace could later enable customers to pay for useful workflow packages or specialist services. That would require licensing, quality review, dispute handling and actual customer demand. It should not reward meaningless jobs merely to manufacture activity. A marketplace proposal is not current creator revenue.

The strategic test is whether a token improves distribution, user participation or coordination enough to justify additional payment friction and operational risk. The service should remain economically understandable without a rising token price. If that test fails, the responsible response is to revise the token design - not to imply that users owe the project speculative support.

16. Draft smart contracts and their limits

Three archived source files describe a possible implementation. **StarloToken** defines the reference fungible token and initial treasury mint. **StarloCheckout** accepts a signed service quote, transfers tokens to a fixed treasury and emits a payment event. **StarloVesting** extends a vesting wallet with an initial cliff. These are draft source files, not an audited or deployed system. Dependency versions, compilation, execution tests, deployment verification and role ownership must be reviewed before use. [C9-C11]

The checkout quote contains a quote identifier, payer, token amount, credit units and expiry. Its EIP-712 domain identifies the checkout application. The contract checks the payer, nonzero values, expiry, unused identifier and recovered authorized signer before transferring tokens. A quote is single-use in that contract's state. EIP-712 standardizes typed-data signing but explicitly does not itself provide replay protection; the application must supply it. [C10; R4]

The checkout owner can change the quote signer and pause or unpause payments. Token and treasury addresses are immutable in this draft. It is not an upgradeable proxy design. These facts do not remove operational trust: a compromised signer could authorize economically harmful quotes, while owner control can interrupt new payments. Administrative roles and all changes would require clear public disclosure.

The draft expects the known fixed token without transfer fees; substituting a fee-on-transfer or unusual token changes the accounting assumptions. An emitted `PaymentReceived` event is evidence of a particular contract

action, not proof that a customer received credits or that AI work was good. The off-chain service must perform independent receipt and account reconciliation.

The vesting source begins linear accrual at its start timestamp and blocks release until the cliff. At the cliff, accrued entitlement can become releasable; the cliff is not a second start date. Tokens funded later follow the same schedule and can already be partly vested. Inherited beneficiary ownership can be transferred. A funded, verified vesting deployment and published schedule are necessary before making a "locked allocation" claim. [C11]

17. Proposed payment-to-credit protocol

This section specifies a proposed flow. **It is not implemented by the website's new Buy StarloOS tab.** The tab remains an informational not-live control until a separately verified launch decision and release activate it.

A customer would first authenticate an application account and link a wallet through a clear, scoped process. A server would issue a short-lived quote identifying the exact network, token contract, checkout contract, payer, token amount, credit amount, expiry and quote identifier. The user would inspect the transaction in their wallet. The application must never request a seed phrase or use a model to approve the payment.

After submission, a payment monitor would verify the transaction on the intended network, the expected contract and event, the payer and amount, and the applicable finality policy. It would bind the quote to the correct application account. A wallet address in an event is not sufficient if the service never established which account owns that purchase.

Credit issuance would occur in a transaction that both records the unique receipt and changes the balance. The permanent identity should include chain, checkout contract, transaction hash and log index, with quote uniqueness tracked independently. Concurrent callbacks must not issue duplicate balances. A browser's "payment successful" message is a user-interface event, not settlement authority.

Quotes should be generated using a funded pricing policy, not an easily manipulated spot price from a thin pool. They need bounded lifetime, explicit fees, minimum/maximum purchase sizes and slippage handling. The operator must reject a quote if execution cannot reasonably fund the promised service. Unsupported networks or tokens should fail closed rather than be treated as accidental discounts.

A production receipt should show the original payment, credit issuance, relevant network evidence, balance change and support path without publishing private mission content. The accounting should reconcile payments, refunds, reversals and service liabilities. A later agent task has its own execution record; payment is not proof of task acceptance.

18. Settlement, exceptions and reconciliation

Situation	Proposed handling	Why it matters
Quote expires before payment	Reject or apply a previously disclosed recovery policy	A stale price must not silently bind new service obligations

Situation	Proposed handling	Why it matters
Wrong network or token	Do not issue automatic credits	Similar tickers and addresses can refer to unrelated assets
Duplicate callback	Return the recorded receipt without new issuance	Retries are normal and must not create value twice
Transaction not final	Keep pending; do not show spendable purchased balance	Chain state may change before the chosen finality condition
Reorganization after observation	Recheck canonical evidence and apply a documented hold/reversal policy	A log observed once is not permanent evidence
Provider timeout during work	Hold uncertain usage for review	Retrying may duplicate paid work or side effects
Refund approved	Record the authorized adjustment and reconcile it with the original receipt	Refunds are not arbitrary negative balances or forgotten events
Token transfer outside checkout	No automatic credit entitlement	A bare transfer may lack an account binding and valid quote
Operator or signer compromise	Pause new purchases, preserve evidence and communicate recovery	Treasury and service solvency can be affected independently

Refund terms require a deliberate choice of valuation basis and timing. Refunding a number of tokens, a quote-denominated service value or an unused credit purchase can produce different results after price changes. The operator must disclose which policy applies and obtain legal review rather than treating token volatility as the customer's invisible problem.

Permanent payment receipts should not be deleted when the interface shortens recent history. They need backups, access controls and restore testing. If the service cannot establish whether a payment has already been credited, it should hold for reconciliation rather than guess. A log index and quote identifier reduce ambiguity but do not substitute for reliable storage.

Operational reconciliation should compare the canonical chain, treasury receipts, credited balances, outstanding reservations and delivered usage. Exceptions need named responsible operators and an auditable resolution record. These procedures remain launch requirements; the private alpha's internal grant ledger is not a complete accounting or compliance system.

19. Service pricing and economic sustainability

A proposed service price must cover variable model charges, paid tool usage, infrastructure, storage, settlement, support attributable to usage and a contribution toward fixed operations. Credits are a way to present a predictable budget; they are not a mechanism for eliminating real costs. An arbitrary token-price target is not a substitute for measuring those costs.

Let `c` be the direct service cost in a common accounting unit, `m` a target contribution margin, and `f` disclosed per-purchase costs not included in C. One illustrative pricing rule is `service price = C / (1 - m) + f`,

provided `0 <= m < 1`. This is a planning identity, not an adopted tariff. A 20% markup on cost is not the same as a 20% margin on revenue.

The following figures are **hypothetical examples only**. They are neither provider quotes nor existing STARLO prices.

Example	Assumption	Implication
Service purchase	1,000 credits priced at $10	Each illustrative credit corresponds to $0.01 of services
Token checkout	Executable price of $0.02 per STARLO	500 STARLO before separately disclosed fees and tax
Later price decline	New executable price of $0.01	A new $10 purchase requires 1,000 tokens; old service credits need not change
Small task	Direct cost $0.04; customer charge $0.05	$0.01 contribution and 20% margin before remaining overhead
Monthly example	1,000 customers spend $20; direct costs total $12,000	$8,000 remains before fixed costs, tax and other obligations, not holder income

Unused credits create future service obligations. A proposed treasury policy would retain enough liquid, usable funding for expected consumption and adverse cost scenarios. The operator should not burn all payment tokens while promising unfunded work. Holding volatile treasury tokens at a quoted market value is not equivalent to possessing cash that can actually pay provider bills.

Pricing also needs controls for models with different rates, long contexts, repeated retries, malicious usage and expensive connectors. External provider spending caps should supplement the application's own ledger. Budgets enforced by the application do not protect against a stolen provider credential used outside it.

20. Token demand, liquidity and valuation boundaries

Usage may generate transaction demand, but the length of time users hold payment inventory matters. An illustrative service with $100,000 of monthly purchases and an average customer inventory of three days' spending implies roughly `$100,000 x 3 / 30 = $10,000` in aggregate transaction inventory. This is not a token valuation model, a market-cap prediction or a statement about actual StarloOS usage. It explains why gross service volume cannot be equated to a large required token valuation.

A treasury that converts received tokens to fund service delivery can create sell flow alongside customer buy flow. Liquidity depth, spread, volatility, ownership concentration, unlocks and speculative activity can dominate a token's price. An economic model must include those interactions rather than count only the buying side.

A future exchange listing does not guarantee that a buyer can sell at an expected price. Pool depth can change, positions can be withdrawn and transaction ordering can disadvantage users. A locked liquidity position, if one is ever used, would need a verified address, terms, unlock date and explanation of what is and is not locked. No such position is asserted here.

The whitepaper provides no target price, valuation multiple, guaranteed floor, expected return or assurance of appreciation. It does not claim that a fixed draft supply is necessarily scarce relative to demand. Users and prospective holders should evaluate the project, actual token rights and market conditions separately.

DEX Screener is a third-party market-data interface whose pair data is identified by chain and pair or token address. Its existence does not establish official endorsement or safety. The website must link only an identity-verified project pair after launch; a generic ticker search can surface unrelated assets. This edition deliberately publishes no pair link. [R5]

21. Allocation, vesting and treasury policy

This paper does not invent an allocation chart. The draft token sends its entire initial supply to one treasury address; it contains no enforcement of percentages for the team, community, liquidity, investors or ecosystem. No identified beneficiaries or funded locks were supplied for this edition. Publishing an attractive pie chart would not establish any of those facts. [C9-C11]

Before a launch, the responsible issuer should publish a complete allocation register: category, amount, percentage, recipient or controlling entity, transfer conditions, vesting contract, initial unlock, ongoing release schedule and any discretionary powers. Allocations must reconcile to the actual issued supply. Treasury holdings, reserved allocations and genuinely circulating tokens should not be counted interchangeably.

Vesting should be described using actual timestamps and mechanics. A cliff may release accrued entitlement all at once; a linear schedule can still create concentrated sell pressure. Later funding and beneficiary transferability matter. The existence of source code is not evidence that tokens have been deposited into a lock, and an administrator's promise is not an on-chain restriction.

A proposed treasury policy would separate operating reserves, service-credit obligations, token inventory and discretionary ecosystem budgets. It should define who can authorize payments, how signers are replaced, which transaction limits require additional review, and how conflicts are disclosed. Multi-signature control is a recommendation, not an implemented or verified custody arrangement in this edition.

The treasury should publish a consistent accounting basis for liquid funds and liabilities. Token balances marked at an illiquid quoted price can overstate available resources. Commercial budgets must not assume future token appreciation or rely on new token buyers to meet old service obligations. Emissions are financing or distribution decisions, not revenue from external customers.

Any future discretionary burn, buyback or incentive program would require a separate decision, funding analysis, legal assessment and accurate disclosure. No such program is promised. Changing those policies must not be presented as an automatic feature of the existing token draft.

22. Governance, administration and accountability

The current software is operator-administered. Owners configure services, allocate pilot credits, invite users, resolve uncertain usage and control deployment. The archived checkout design includes administrative roles for pausing payments and changing the quote signer. No token-holder governance module, voting system, decentralization milestone or binding community control is implemented. [C2, C10]

A future governance proposal would first need to specify what decisions are eligible, who can participate, how voting power is calculated, how proposals are executed and which emergency powers remain with operators. A poll is not on-chain governance. Token ownership alone does not establish legal authority over a company or entitlement to override safety and compliance obligations.

Governance design must address concentration, delegated voting, bribery, borrowed voting power, low turnout and conflicts between service users and speculators. It should not enable arbitrary transfers of customer funds or publication of private user data. Technical controls and legal responsibilities must remain coherent even when a community vote is popular.

For the near-term product, the accountability priority is simpler: name a responsible operator, publish support and security contacts, maintain change records, disclose administrator powers and verify releases. No actual person's identity, corporate registration, office location or professional credential is invented in this paper. These disclosures remain required before a credible public paid launch.

23. Optional creator and service economy

A potential marketplace could allow authors to sell maintained workflow packages, reference sets they have rights to distribute, or bounded specialist services. The customer would need to understand whether they are buying a reusable template, hosted execution, a human-reviewed deliverable or access to an external integration. Those products have different costs and responsibilities.

The proposed marketplace should define authorship and licensing, quality checks, version pinning, fees, refunds, abuse reporting and dispute resolution. A successful test on one model is not certification for every provider. Workflows that can send data to outside services need visible permission summaries and source review. A signed package can establish origin but cannot establish correctness or absence of malicious behavior.

Creator compensation should come primarily from genuine customer purchases. Rewards for raw job count, template submissions or token volume can encourage low-quality activity and circular trading. Any subsidies should be separately identified, capped and excluded from organic-revenue metrics. A grant is not a customer payment.

No marketplace, fee split, creator income stream or token reward schedule is active in this edition. The concept should be developed only after the core execution and payment paths work reliably. It is not a justification for a token sale before the product demonstrates useful demand.

24. Security model and operational controls

The relevant threat actors include malicious website authors, compromised model or connector services, hostile guests, careless authorized users, malicious installers, stolen operator credentials and compromised settlement roles. Security must cover the complete path, not only the model prompt. A safe-looking interface does not protect an unsafe backend.

The source uses capability checks, exact-action approvals, mission file namespaces, authentication and session controls, private-network restrictions for web access, bounded inputs and outputs, and explicit treatment of uncertain external calls. These are layered controls, not proof that no exploit exists. A model can still generate harmful advice, misinterpret evidence or persuade a user to approve a dangerous action. [C1-C3, C12]

Indirect prompt injection occurs when external content influences a model's instructions or actions. It can cause disclosure or misuse of tools; adding retrieved documents or fine-tuning does not eliminate the problem. The proposed defense is to combine minimal permissions, source separation, restricted destinations, explicit approvals and review of unexpected requests, not to rely on a stronger instruction sentence alone. [R6]

The optional local Python tool uses a fixed Docker invocation with no network, no host mounts, read-only root, an unprivileged user and resource limits. The operator must configure Docker and a suitable pre-pulled image. Host-kernel and daemon vulnerabilities remain possible, and this implementation is not a certified sandbox. Arbitrary remote shell operation is not granted through the Vercel API. [C3, C12]

Provider keys and treasury keys serve different roles and must never share a model context. The proposed payment signer should be isolated from the agent runtime, and treasury signing should require independent control. Neither a source document nor an MCP result should ever be accepted as authorization to export a credential or sign a financial transaction.

The release process must avoid publishing runtime data, credentials, debug logs with secrets or authentication state. A release checksum supports integrity comparison, not publisher identity. Signed installers, reproducible source mappings, a dependency review and native-platform acceptance remain important distribution requirements. The existing Windows candidate is not upgraded to a signed release by this paper.

25. Privacy, data lifecycle and user control

Local storage describes where the application keeps its records; it does not describe where a configured model runs. User notes, task instructions, approved memories, selected documents and tool results may be sent to the chosen inference provider. Search and RPC requests reveal the query or address to those service operators. Connector requests follow the configured connector's own behavior and terms.

The desktop provider settings include in-memory use and an optional passphrase-encrypted credential vault. The workspace database itself is not described as end-to-end encrypted. Operating-system disk protection, device access control and backups remain the user's responsibility. Forgetting a vault passphrase or losing local files can affect access; an undocumented recovery promise should not be made. [C7, C12]

A production operator needs a documented data inventory: categories collected, purpose, lawful basis where required, retention periods, recipients, cross-border transfers, security measures, user rights and complaint contacts. Those details depend on the actual operator and providers. The supplied pilot privacy page is not a substitute for completing them.

Deletion requires care. Removing a visible run may not erase a provider's records, a user's exported ZIP or required payment/accounting evidence. Conversely, financial-record retention does not justify retaining private prompts indefinitely. The proposed policy should minimize sensitive payloads in permanent receipts and distinguish transaction metadata from user content.

No document, prompt, API key or private memory should be written to a public blockchain as part of the proposed payment flow. Even hashing private content can create linkage or disclosure risks if the content is guessable. Public payment data may also connect a wallet to a service relationship; account design should avoid unnecessary public identification.

26. Performance, reliability and scaling boundaries

The current source applies explicit pilot limits, including a small number of active missions, bounded model turns and tool actions, limited retained runs, and a serialized workspace ceiling near 950 KB. Such limits reduce uncontrolled work but are not a scaling architecture. A large team or long-running workload can reach them quickly. [C1]

The hosted function configuration has a finite execution window. Each advance request performs a bounded step. A separate dispatcher can arrange more steps; it does not remove the need for durable scheduling, retries, ownership, backpressure and exception handling. Local background execution lasts only while the runtime is running and the machine is available.

The proposed scale-up path separates accounts, missions, ledger entries, receipts and files into appropriate storage structures; uses object storage for larger artifacts; introduces durable queues and dead-letter handling; and validates tenant isolation and restore behavior. Capacity should be measured under representative workloads before increasing public concurrency limits.

Reliability metrics should include task queue delay, action latency, provider error rates, approval wait time, cost overruns, recovery time and lost or duplicated transitions. A frontend frame-rate counter measures rendering, not AI throughput, customer satisfaction or backend health. Synthetic visual instruments are never a substitute for an operational monitoring system.

Load testing must include failures: provider throttling, database conflicts, duplicate callbacks, expired approvals, worker termination, malformed tool results, storage saturation and interrupted exports. No throughput, uptime or latency guarantee is established by the current UI or this document.

27. Evidence, evaluation and transparent reporting

A useful product dashboard distinguishes attempts, finished executions, structurally complete outputs, accepted outputs, rejected outputs, refusals and unresolved runs. Combining these into one "successful jobs" number conceals important failures. Human acceptance also needs a defined meaning: a clicked button may reflect satisfaction, not independent truth verification. [C4]

The proposed reporting set is: accepted jobs by task type; repeat paying users; cohort retention; direct cost per accepted job; refunds and disputes; median and tail completion time; provider failure rates; and external paid receipts reconciled to usage. Subsidized or incentive-driven activity should be reported separately from organic purchases.

Security and product tests should disclose their scope. Fixtures establish repeatable behavior under controlled inputs. Live-provider tests establish behavior for a particular configuration and moment. Native-platform tests establish installation behavior on the tested platform. None alone proves all other layers. This edition does not convert earlier test counts into an independent audit badge.

Any public benchmark should publish a task set or sufficient methodology, a frozen configuration, scoring criteria, exclusions, failed cases and the date. Evaluation should include sources that are unavailable or contradictory, prompts that ask for prohibited actions, stale knowledge and intentionally missing deliverables. Reporting only attractive examples creates an unreliable impression of capability.

The whitepaper's source manifest is a reproducibility aid, not proof that the current live website uses the same code. Operators should publish release identifiers, downloads, hashes, changes and known issues together. A new paper edition should not silently overwrite an older token or commercial policy.

28. Risk register

Risk	Potential consequence	Control or proposed response	Residual limitation
Model hallucination	Incorrect or unsupported result	Sources, explicit uncertainty, human acceptance	Reviewers and sources can also be wrong
Prompt injection	Data disclosure or unwanted tool request	Scoped tools and exact-argument approval	Human approval and model judgment remain fallible
Provider outage or price change	Failed work, delays or higher cost	External caps, bounded runs, visible failures	No promised service continuity across providers
Lost or compromised credentials	Unauthorized usage or data access	Minimal access, secret isolation, rotation	Compromised user devices remain a risk
Operator compromise	Manipulated service, accounts or receipts	Segregation of roles and independent review	Central service administration remains trusted
Unsafe download	Device compromise or failed installation	Hashes, signing and native acceptance gates	Current Windows candidate is unsigned
Storage failure	Lost state or inconsistent accounting	Atomic updates, backups and restore testing	Pilot storage is not a full production finance ledger
Token-price volatility	Purchase cost changes or treasury shortfall	Short quotes and funded service reserves	No guaranteed liquidity or stable price
Illiquid or manipulated market	Unexecutable price and harmful trades	Executable pricing, limits and disclosure	Market conditions can change rapidly
Concentrated supply or unlocks	Control imbalance and sell pressure	Verified allocations and schedules	No final allocation or funded lock is established
Contract or signer flaw	Loss, improper quotes or stopped payments	Testing, independent audit and role isolation	Draft contracts are not audited deployments

Risk	Potential consequence	Control or proposed response	Residual limitation
Chain or RPC failure	Incorrect pending status or delayed credit	Chain binding, finality checks, reconciliation	Finality and provider assumptions still apply
Fake token or website	Funds sent to an impersonator	Verified CA, chain and official links	Names, tickers and screenshots are easily copied
Regulatory change or classification	Restrictions, redesign or discontinuation	Jurisdiction-specific legal review	A utility label does not resolve classification
Insufficient demand	Service cannot sustain its costs	Paid pilot and retention evidence	Product-market fit is not established
Project discontinuation	Lost future service utility	Wind-down policy and reserves	No guarantee of redemption or uninterrupted operation

Token holders could lose all the value they commit to a future token. A service could fail even if its software is open source. Source availability does not recreate the operator's treasury, customer support, integrations, infrastructure or legal permissions. Neither token ownership nor a visually advanced interface reduces these risks.

29. Legal, issuer and disclosure readiness

This is a technical and economic design paper, not a claim of regulatory compliance. The identity and jurisdiction of the responsible issuer or service operator have not been supplied. Applicable obligations depend on the actual rights, distribution, marketing, customer locations and activities. Qualified counsel must assess them before a public launch.

For offers within MiCA's relevant scope, Article 6 describes required information about responsible parties, the project, rights, technology, risks and environmental matters, and restricts assertions about future token value. This paper does not supply an authorized management-body statement or claim an authority approved its contents. [R7]

Article 7 requires relevant marketing to be fair, clear and not misleading and consistent with the whitepaper; where a whitepaper is required, timing restrictions apply. Article 4 contains conditional exemptions, with implications when admission to trading is intended. A "utility token" label is therefore not a blanket exemption. [R8-R9]

A formal offering document, where required, needs the applicable prescribed structure, notices, machine-readable format, responsible approvals and notification/publication process. A website article and this PDF do not automatically satisfy those requirements. No regulatory filing, registration, license, approval or exemption is represented as completed.

Before launch, the operator must also assess consumer terms, complaints and refunds, tax treatment, record retention, sanctions and AML obligations where applicable, provider contracts, intellectual property, data protection and restrictions on investment or financial advice. This section is a readiness list, not legal advice for a particular issuer or buyer.

The project should avoid claims such as guaranteed returns, assured listings, risk-free credits, unlimited funded training, certified contract safety or independent audit approval without evidence. Commercial commitments and marketing statements must be reconciled with what the code and operating resources can deliver.

30. Environmental and resource considerations

The project has not selected a deployment chain or measured an operating footprint for a token service. It therefore makes no carbon-neutral, carbon-negative or energy-efficiency claim. The visual interface's apparent complexity is not a measurement of compute or environmental impact.

A responsible assessment would separate local device energy, model inference, hosting, storage, optional tool execution and any blockchain transactions. The system should not attribute the entire energy use of a network to each transaction, nor assume that an advertised consensus mechanism eliminates all impact. Methods and boundaries need explicit documentation.

The current interface includes reduced-motion support and Full, Eco and Pause modes. These are usability and rendering controls, not quantified environmental offsets. A future public sustainability disclosure should use measured or appropriately sourced estimates, identify uncertainty and comply with applicable reporting requirements rather than attach an unsupported green badge.

31. Delivery roadmap and release gates

Gate	Required evidence before advancing	What is not enough
A. Local product acceptance	Clean installation, real model task, denied approval, export, persistence and shutdown	A compiled executable or passing fixture tests alone
B. Hosted private pilot	Configured access and storage, funded live calls, bounded work, recovery and support	A READY website deployment with no runtime configuration
C. Repeat customer value	Non-incentivized paid jobs, repeat use, accepted outputs and measured costs	Airdrop farming, generated files or social impressions
D. Payment testnet	Correct quotes, replay prevention, canonical receipts, credits, refunds and exception recovery	A token deployed without the service bridge
E. Independent review	Security assessment, contract audit, role review and native distribution checks	Self-reported test counts described as an audit
F. Legal and commercial readiness	Identified issuer, required disclosures, policies and approvals	A generic disclaimer or a draft whitepaper
G. Mainnet service launch	Verified contracts and roles, funded operations, monitored settlement and official links	A website countdown, announced ticker or unverified pair
H. Expanded ecosystem	Proven marketplace demand, quality controls and sustainable benefits	Emissions described as organic revenue

These gates are proposed sequencing, not fixed delivery dates or contractual promises. Some work can proceed in parallel, but a token launch should not bypass a failed payment or legal gate. A missed gate should be reported openly with its consequence for the product and token plans.

A practical pilot might use 20-50 external testers and at least 100 non-incentivized paid jobs before a broader launch decision. Those numbers are suggested experiment parameters, not an industry standard, existing traction, statistical assurance or promise of success. The decisive evidence is whether users return for useful work at a sustainable cost.

No date for mainnet launch, exchange trading, signed installers, native Mac distribution, governance, staking or a marketplace is promised in this edition. Publishing a roadmap item does not create a present token entitlement.

32. Public launch status, navigation and anti-impersonation

The website additions intentionally display four distinct controls: **Whitepaper**, **DEX Screener - Not live**, **Buy StarloOS - Not live**, and **CA - Not live**. The latter three are informational status controls. They do not request a wallet connection, copy a placeholder address, select a random token with the same ticker or send a visitor to an unverified purchase page.

A safe official-link release needs a chosen network and chain identifier, deployed contract address, verified source/build details, known administrative roles, independently checked pair identity and an approved purchase destination. Those facts should be reviewed together by responsible operators. A ticker, screenshot, direct message or community post is not enough.

CA means contract address. There is no usable official CA in this release. The paper does not assert that nobody anywhere has created a token called STARLO; it states that this release does not publish or verify one as the official project asset. Visitors should not infer legitimacy from a matching name.

Before any future purchase, a user should compare the exact network and address with the current verified project disclosure and inspect the transaction in their wallet. A connection request is not payment verification. A website should never ask for a seed phrase or suggest disabling wallet warnings to complete a purchase.

A later launch should ship a reviewed change to the navigation and status manifest, retain the previous paper for reference and publish a change record. Editing "not live" to "live" is not sufficient to implement payments, satisfy legal obligations or establish that a pair is liquid.

33. Open decisions and accountability register

Unresolved item	Why it is material	Evidence needed to close it
Issuer and operator identity	Determines responsibility and legal obligations	Authorized legal name, jurisdiction and contact disclosures

Unresolved item	Why it is material	Evidence needed to close it
Final token network	Determines addresses, finality, gas and user risks	Approved network decision and tested deployment configuration
Final supply and distribution	Determines concentration and dilution	Approved schedule reconciled to verified contract behavior
Allocation and vesting	Determines when and by whom tokens can be sold	Beneficiaries, funded contracts, dates and control analysis
Liquidity arrangements	Determines market access and execution risk	Verified pools, positions, controls and disclosures
Credit tariff and refunds	Determines customer obligations and sustainability	Versioned pricing, measured costs and reviewed consumer terms
Treasury reserve policy	Determines ability to provide prepaid service	Funded accounts, liability methodology and reconciliation
Windows readiness	Determines whether users can install safely	Native-platform evidence and signed publication decision
Live inference and integration reliability	Determines real task capability	Repeatable live-provider acceptance records
Independent security review	Determines external assessment of risks	Named review, exact scope, version and remediation record
Regulatory documentation	Determines lawful offer and marketing process	Jurisdiction-specific counsel review and required filings
Creator or membership benefits	Determines any additional token utility	Explicit rights, sustainable costs and implemented mechanisms

This register is intentionally incomplete in facts rather than filled with fictional corporate names, exchange logos or lock guarantees. Missing material disclosures should block the affected launch claim. Operators must update the register as evidence exists, not retrospectively rewrite the history of what was known.

34. Frequently asked questions

Is StarloOS a replacement for Windows or macOS?

No. "OS" is product branding for an AI-agent workspace. The delivered local software runs on an existing computer operating system. It is not a bootable disk image or a hardware operating system.

Does the downloadable product include free AI forever?

No. No prepaid inference or pretrained model is bundled. A configured external model account or separately installed compatible local model is required. Local inference can also consume device resources and needs suitable hardware and model rights.

Do users have to buy STARLO to use local software?

The current bring-your-own-model path does not require token payment. The proposed managed service would sell execution credits. Open-source local code is not a reliable place to enforce access to infrastructure the operator does not provide.

Can agents be trained?

They can be configured with instructions, approved examples, reference material and approved memory. Academy practice helps inspect results. That is not model-weight training or a guarantee of competence, and literal phrase checks are not semantic certification.

Does a completed mission mean it is correct?

No. The system separates execution status from user acceptance, and structural checks can miss substantive mistakes. Sources, assumptions and consequential outputs need review.

Is STARLO live or available to buy from this website?

Not in this edition. There is no verified official CA, pair or purchase destination, and the token-to-credit bridge is not activated. The status tabs explain this without initiating a transaction.

Does STARLO grant equity, dividends or guaranteed yield?

No such rights are implemented or promised here. A later token design must disclose exact rights and receive applicable legal review. Company activity and token price are different matters.

Why not promise burns, staking rewards or a price floor?

Those mechanisms do not create real customer demand or pay service bills by themselves. They can introduce obligations and risks. None is necessary to explain the initial proposed service utility, and none is active in this release.

Is the product fully decentralized?

No. Local user control is different from decentralized infrastructure. Hosted accounts, providers, databases and administration remain trusted dependencies. The draft token cannot decentralize a service merely by existing.

Where can a user verify downloads and claims?

Use the website's release records and checksums, read the known limitations and compare versions. Checksums establish matching bytes, not absence of malware or publisher identity. This paper's source manifest identifies the reviewed baseline, not a live-deployment certification.

Appendix A. Capability and dependency checklist

Capability	Minimum dependency	Scope of this edition
Website and decorative instruments	Modern browser and static hosting	Public interface; no real-task activity implied
Local saved workspace	Supported Node runtime or candidate installer	Source implementation; native installer acceptance unresolved
Agent inference	Compatible configured provider or local model	No bundled model or funded account
Public web read	Accessible public HTTPS page and approval	Text extraction only; bounded retrieval
Web discovery	Configured Brave key and approval	External paid service dependency
Mission files	Granted file capability and applicable approval	Virtual mission namespace; bounded text files
Chain snapshot	Configured supported RPC endpoint and approval	Read-only public data; no wallet or price oracle
MCP tools	Owner-configured server and compatible transport	External tools may have side effects; exact-call approval
Python	Local owner, explicitly enabled Docker and image	Restricted execution flags; no production sandbox certification
Hosted persistence	Database credentials and owner authentication	Pilot architecture with size/concurrency limits
Unattended hosted work	Properly configured dispatcher	Not created by the static website alone
STARLO checkout	A future verified settlement and credit bridge	Proposed only; not live

Appendix B. Mission and payment invariants

The following are implementation properties or proposed acceptance criteria, as indicated. They are useful review targets, not a claim of formal verification.

Mission scope: a requested tool must be permitted by the mission's captured capabilities and current runtime availability. A model-generated tool name is not executable merely because it is syntactically valid. **Approval identity:** approval covers the recorded call and exact arguments; rejected or changed requests do not inherit consent. **Action ownership:** only the matching active claim may settle a step. **Accounting:** balances and reservations use integers, and settlement cannot exceed an authorized budget without a separate decision. **Outcome:** structural completion and human acceptance remain distinct. These are implemented design properties to keep under regression testing. [C1-C4]

Payment identity: the canonical receipt must match the selected network, checkout address, quote, payer, amount and account binding. **Single issuance:** one verified purchase produces at most one credited balance transition, even after retries or display-history pruning. **Finality:** a pending observation is not a spendable

balance. **Recovery:** uncertainty enters a documented hold rather than an invented success. **Isolation:** agents cannot sign quotes, authorize refunds or access treasury keys. These are proposed payment acceptance criteria; a production bridge has not been implemented. [C10]

A token contract cannot enforce every off-chain obligation. A successful on-chain transfer can coexist with a service outage, lost account mapping or insolvent operator. The customer must be told which component guarantees which property and what remedy, if any, applies when it fails.

Appendix C. Illustrative end-to-end research mission

This is a proposed customer acceptance scenario, not a report of a completed live task. A user creates a watchlist item, selects three official public sources and adds an earlier report as a reference. The task requests `brief.md` and `sources.csv`, a comparison period, explicit uncertainty and no financial transactions.

Scout proposes reading the exact URLs. The user reviews each destination and approves only the intended requests. Retrieved text is treated as source material, not new operating instructions. If a page is unavailable, Scout records that limitation rather than fabricating content. Prism compares claims and calculations. Quill writes the requested files within the mission namespace. Gate identifies unsupported statements and missing coverage without presenting its review as a certified audit.

The runtime records attempted tools, approvals, usage and file creation. It checks that required deliverables exist and are nonempty, then presents the output for human review. The user may accept or reject it, noting the reason. Any provider consumption remains visible regardless of that review decision. A future paid service would apply its disclosed refund policy separately.

The user downloads the report and table, restarts the session, and verifies that the saved agent and permitted records remain available. A repeat run uses a new budget and the previously accepted brief as a clearly identified baseline. Testing this path with real sources and a funded provider is a release requirement, not a promise supplied by a UI animation.

Appendix D. Contract, ledger and failure test plan

A future contract release should pin compiler and dependency versions, compile deterministic artifacts, test constructor validation and supply, inspect inheritance, and verify the deployed runtime bytecode. Checkout tests should cover wrong payer, altered fields, expired quote, duplicate quote, wrong domain or chain, invalid signer, paused state, signer rotation and unexpected token behavior. Vesting tests should cover before cliff, at cliff, intermediate accrual, completion, later funding and ownership transfer.

The off-chain bridge should test duplicate and reordered callbacks, delayed finality, a changed block hash, inconsistent providers, missing logs, wrong network, wrong contract, account mismatch, payment observed before database availability, receipt written without balance mutation and balance mutation without receipt. The last two conditions must be prevented by transactional design, not repaired by optimistic customer-service guesses.

Reconciliation tests should restore a backup and reprocess known payments without issuing more credits. Refund tests should retain the original purchase linkage and pricing basis. Security tests should verify that agent

inputs cannot alter quote signer configuration, trigger treasury transactions or bypass customer-account authorization.

A successful testnet demonstration remains different from mainnet readiness. It should be followed by independent review, operational rehearsal, role verification and legal checks. Neither the number of tests nor a deployed token address establishes completion of this plan.

Appendix E. Glossary

Term	Meaning in this paper
Agent	A model-driven process with instructions, context and a constrained tool set
Mission	A bounded task with an execution record, permissions and budget
Workflow	An ordered reusable sequence of specialists in the current product
Tool call	A model request that application code validates and may execute
Approval	User authorization for a recorded action and its exact arguments
Acceptance	A user's review decision about a finished output, separate from execution status
STARLO	Proposed blockchain service token; not live in this release
Service credit	Internal accounting unit for authorized usage, not a second cryptocurrency
Model token	Provider measurement of input/output usage, unrelated to the STARLO ticker
CA	Contract address, meaningful together with its network
EVM / ERC-20	Execution environment / common fungible-token interface family
Quote	Proposed signed purchase terms with amount, payer, identifier and expiry
Receipt	Durable record of observed and reconciled payment evidence
Finality	The selected network-dependent condition before payment issuance is treated as sufficiently settled
MCP	Protocol family used by the optional configured tool-server integration
BYOK	Bring your own key or model-provider access
Fine-tuning	Training that changes model weights; not implemented by Academy examples
Vesting	Rules governing when allocated tokens become releasable
Liquidity	Ability to transact at executable prices, not an assurance of resale value

Term	Meaning in this paper
Treasury	Controlled project resources; separate from customer service obligations

Appendix F. Source register and provenance

Bundled implementation references

- **C1 - Execution architecture:** `server/agent-runtime.mjs`, `server/agent-provider.mjs`, `docs/ARCHITECTURE.md`.
- **C2 - Access, accounting and persistence:** `server/domain.mjs`, `server/handler.mjs`, `server/shared.mjs`, `server/storage.mjs`, `server/local-store.mjs`.
- **C3 - Tools and permission boundaries:** `server/agent-tools.mjs`, `server/web-tool.mjs`, `server/mcp.mjs`.
- **C4 - Outcomes and human acceptance:** `server/quality.mjs`, `tests/audit-080.test.mjs`.
- **C5 - Crypto research:** `server/watchlist.mjs`, `server/chain.mjs`, `public/crypto-ui.js`.
- **C6 - Teaching and practice:** `desktop/teaching.mjs`, `desktop/api.mjs`, `public/academy.html`.
- **C7 - Local runtime and providers:** `desktop/run.mjs`, `desktop/provider.mjs`, `server/local-provider.mjs`, `docs/DESKTOP.md`.
- **C8 - Release and hosting:** `scripts/build.mjs`, `scripts/package.mjs`, `scripts/release-files.json`, `release-inputs/desktop-release.json`, `vercel.json`, `public/build-info.json`.
- **C9 - Token draft:** `archive/token-contracts/src/StarloToken.sol`.
- **C10 - Checkout draft:** `archive/token-contracts/src/StarloCheckout.sol`.
- **C11 - Vesting draft:** `archive/token-contracts/src/StarloVesting.sol`.
- **C12 - Security and production limits:** `docs/SECURITY.md`, `docs/PRODUCTION.md`, `docs/SCHEDULING.md`, `VALIDATION.md` and retained release verification files.

The software baseline is the supplied `StarloOS-0.8.1-Hyperdrive-Vercel.zip`. Its SHA-256 and selected unmodified source hashes are recorded in `whitepaper/source-manifest.json` accompanying this paper. This edition adds documentation and navigation; it does not deploy a token, revise the underlying runtime, replace the Windows binary or certify a live service. Historical test reports remain historical reports; new documentation checks are recorded separately.

External primary references

External references were consulted on 22 October 2026. They explain standards and legal or technical context, not endorsement of StarloOS. The protocol and economic recommendations in this paper are the project's proposed design, not requirements claimed to come from these sources.

- **R1 - OpenAI, Function calling:** https://developers.openai.com/api/docs/guides/function-calling
- **R2 - OpenAI, Supervised fine-tuning:** https://developers.openai.com/api/docs/guides/supervised-fine-tuning
- **R3 - Ethereum.org, ERC-20 token standard:** https://ethereum.org/developers/docs/standards/tokens/erc-20/
- **R4 - Ethereum Improvement Proposals, EIP-712:** https://eips.ethereum.org/EIPS/eip-712

- **R5 - DEX Screener, API reference:** https://docs.dexscreener.com/api/reference
- **R6 - OWASP Gen AI Security Project, Prompt injection:** https://genai.owasp.org/llmrisk/llm01-prompt-injection/
- **R7 - ESMA, MiCA Article 6:** https://www.esma.europa.eu/publications-and-data/interactive-single-rulebook/mica/article-6-content-and-form-crypto-asset
- **R8 - ESMA, MiCA Article 7:** https://www.esma.europa.eu/publications-and-data/interactive-single-rulebook/mica/article-7-marketing-communications
- **R9 - ESMA, MiCA Article 4:** https://www.esma.europa.eu/publications-and-data/interactive-single-rulebook/mica/article-4-offers-public-crypto-assets-other

Appendix G. Document control

v1.0 - 22 October 2026. First consolidated technical, product and token-economy whitepaper for the Hyperdrive 0.8.1 baseline. Includes explicit implemented/proposed boundaries, a pre-launch token status, source references, economic illustrations, risk register and acceptance gates. No prior PDF or legally filed offer document is represented as superseded.

The website retains the original application version and existing desktop version. The whitepaper has its own version so a documentation release cannot be mistaken for a new executable or activated payment system. The PDF, web text and Markdown originate from the same document content. Format changes do not alter the status disclosures.

An official later revision should identify its responsible approving entity, changes to token rights or supply, new evidence, withdrawn statements and the status of unresolved launch decisions. No author, reviewer, advisor, auditor or management approval is fabricated in this edition.

End of whitepaper. Token status: not live. Contract address: not published.