OFFPEAK

Product overview and specification

Idle GPUs. Open models. Lower prices when capacity is available.

Version 1.0 · October 12, 2026 · Proposed product, not a report of an operating service

Companion documents: Build document · Internal sourcing and claim review

1. Executive Summary

OFFPEAK is a proposed inference service that turns spare GPU capacity into lower-cost calls to open-weight models. Developers and agents see a live price, authorize a capped charge with a wallet, receive a model response, and get a signed usage receipt whose commitment is later recorded on Robinhood Chain.

The product promise is concrete: direct, authorized compute supply; prices that fall when capacity is underused; no OFFPEAK API key; withdrawal of unused, unreserved paid balances; and independently checkable receipt inclusion on-chain.

Suggested external description: "OFFPEAK makes spare GPU capacity available to AI agents. Run open models at prices that respond to available capacity, authorize each call with your wallet, withdraw unused funds, and verify your billing receipts against on-chain records." Use future tense until these capabilities ship.

The strongest differentiation is the supply relationship and cost structure. OpenRouter's August 31, 2026 terms prohibit resale of model API access. This supports avoiding that dependency; it does not establish that a competitor breaches its own agreement. OpenRouter terms, §7.4

Orbio describes managed OpenRouter access and non-cash-redeemable credits. Surplus currently documents a wider marketplace, including keyless payments and withdrawable USDC deposits; these features cannot credibly be claimed as unique to OFFPEAK. Orbio buyer FAQ · Surplus developer reference

$OFPK is a proposed ecosystem token name. Issuance, allocation, utility, and launch are undecided. P0 does not require holding it. Stablecoin payment keeps inference budgets independent of its market price.

2. Problem

GPU owners can have spare capacity in specific operating windows, while inference buyers have cost-sensitive workloads that can tolerate variable availability. OFFPEAK aims to connect those two conditions. Neither the amount of accessible idle supply nor the achievable discount has been demonstrated for this project.

The economic test is whether spare capacity can produce useful outputs cheaply enough after serving overhead, payment operations, failed jobs, and supplier compensation. A discounted GPU hour alone does not establish cheap inference.

3. Target User

Primary buyer: a developer running extraction, classification, summarization, or other repeatable text workloads that pass an open-model quality threshold and can retry later when capacity is unavailable.

Secondary buyer: an autonomous agent with a dedicated funded wallet and a strict per-call spend cap.

Initial supplier: OFFPEAK-operated hardware or one contracted operator granting explicit permission to serve third-party inference. Public, anonymous GPU onboarding is deferred.

Operator: a small team managing capacity, incident handling, treasury reconciliation, and receipt publication.

4. Goals / Non-Goals

P0 must deliver one paid inference loop end to end: discover → quote → sign → reserve → execute → charge → receive proof → withdraw remainder.

P0 excludes proprietary model relays, anonymous suppliers, model training, GPUs rented directly to customers, multi-chain bridging, fiat deposits, streaming, token trading, yield, staking, and cryptographic proof of correct inference. Buyers purchase inference measured in tokens; suppliers sell access to compute capacity. GPU seconds are operational telemetry, not the customer billing unit.

5. Feature List

ID	Priority	Feature	Flow	Acceptance	Phase
F-001	P0	Versioned model catalog and current capacity	UF-001	AC-001	PH-04
F-002	P0	Capacity-based prices and expiring capped quotes	UF-001	AC-002	PH-04
F-003	P0	Wallet identity and signed per-call authorization	UF-002	AC-003	PH-02, PH-06
F-004	P0	Stablecoin deposits and atomic balance reservations	UF-002	AC-004	PH-03, PH-05
F-005	P0	Metered text inference on vetted GPUs	UF-002	AC-005	PH-04, PH-06
F-006	P0	Withdrawal of available paid balances	UF-003	AC-006	PH-07
F-007	P0	Signed receipts and on-chain batch commitments	UF-004	AC-007	PH-08
F-008	P0	Customer console, developer quickstart, verifier	UF-001-004	AC-008	PH-09
F-009	P0	Operator capacity, reconciliation, and incident tools	UF-005	AC-009	PH-10
F-010	P1	Streaming, tool calls, structured-output profiles	Deferred	Deferred	Deferred
F-011	P1	Standard x402 direct-payment integration	Deferred	Deferred	Deferred
F-012	P1	Queue-until-cheaper jobs; delegated wallet budgets	Deferred	Deferred	Deferred
F-013	P2	Supplier self-service and multiple regions	Deferred	Deferred	Deferred
F-014	P2	$OFPK utility and stronger execution attestations	Deferred	Deferred	Deferred

6. User Flows

UF-001 - Find capacity. Open the catalog; inspect model revision, limitations, region, live input/output rates, capacity timestamp, and recent availability. Enter a prompt and output limit. Receive a quote valid for 30 seconds. If capacity is stale or exhausted, show unavailable; do not invent a discount.

UF-002 - Fund and call. Connect a supported EOA wallet, sign a login challenge, and transfer the supported token from that wallet to the configured treasury. Wait for finality before balance becomes spendable. Sign a request with the quote, body hash, maximum cost, expiry, and unique nonce. Reserve the maximum atomically; execute; debit actual measured usage; release the remainder; return response and receipt. On an unsuccessful P0 response, release the reservation and charge zero.

UF-003 - Withdraw. Show available, reserved, and pending amounts separately. Sign a withdrawal to the same connected wallet. Reserve the amount once; broadcast; expose transaction status; mark paid after finality. Pending transactions remain reserved. Withdrawing from OFFPEAK to a wallet on Robinhood Chain is not a withdrawal through the bridge to Ethereum.

UF-004 - Verify. Download a private receipt, salt, operator signature, Merkle proof, and chain reference. Verify locally. Show pending, published, and finalized status separately. Verification proves commitment and signature integrity; it does not establish truthful metering, genuine external demand, or execution correctness.

UF-005 - Operate. Approve a contracted worker; record image/model/license evidence; inspect health; disable new work on degraded capacity; reconcile treasury to liabilities; publish receipt batches; resolve incidents with immutable audit entries.

7. UX/UI Direction

Lead with model, current price, availability, and maximum authorized cost. The primary CTA is **Run a request**. The secondary CTA is **Connect an agent**. Keep token speculation outside the inference flow.

Account balances always show the token symbol and network. Explain that "no API key" means wallet authorization, not anonymous or unrestricted access. Describe P0 as a custodial prepaid balance: OFFPEAK controls treasury funds and processes withdrawals.

8. Design System Direction

Use a restrained infrastructure-console style: near-black ink, warm white surfaces, one teal accent, tabular numerals for money and usage, and readable monospace for request examples. Use 4/8/16/24/32 px spacing and 44 px minimum touch targets.

States are textual as well as colored: Available, Limited, Unavailable, Funds pending, Reserved, Receipt pending, Receipt finalized. Show price changes with timestamp and units. Avoid artificial "live" counters or charts built from fabricated activity.

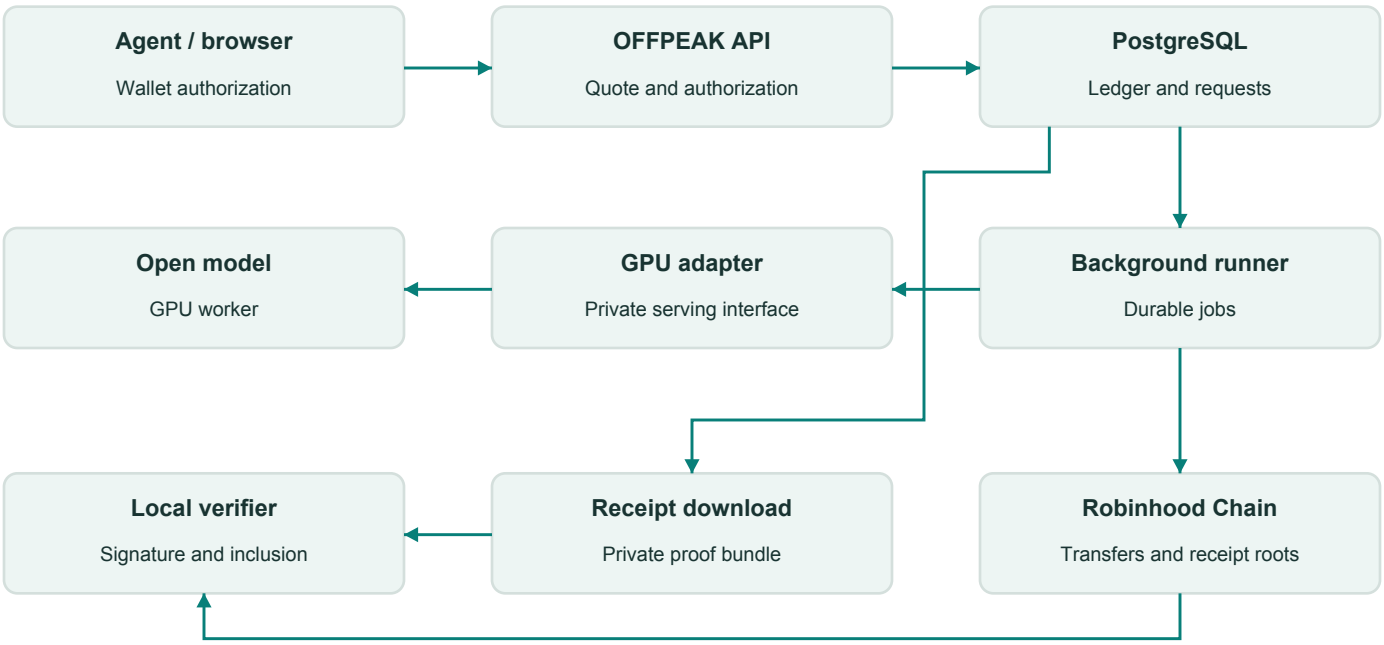
9. Information Architecture

ID	Route	Purpose
RT-001	`/`	Product explanation, available models, truthful claims
RT-002	`/models`	Model catalog, rates, capacity, limitations
RT-003	`/app`	Wallet, balance, recent requests, funding
RT-004	`/app/requests/:id`	Status, measured usage, receipt download
RT-005	`/app/withdraw`	Amount, destination, status and fee disclosure
RT-006	`/verify`	Local receipt verification; optional public root lookup
RT-007	`/docs`	Supported API subset, wallet client, payment semantics
RT-008	`/ops`	Operator-only capacity and financial controls
RT-009	`/status`	Service, capacity, withdrawals and anchoring status

10. Technical Architecture

Use a TypeScript repository containing one Fastify control application, a React/Vite frontend served by that application, PostgreSQL, a small background runner sharing the same domain code, a private GPU adapter, and a minimal Solidity receipt registry. The background runner is justified by durable dispatch, chain indexing, withdrawals, and receipt publication.

The existing "open-model tech" is an unverified integration boundary. Use an adapter contract so it can replace a baseline vLLM worker without changing billing. vLLM supplies an OpenAI-style serving interface and operational metrics; it does not supply OFFPEAK's payment or proof system. vLLM serving documentation · Metrics



No request waits for a receipt-root transaction. Deposits and withdrawals use chain finality; calls debit a prepaid off-chain ledger. The chain does not execute the model.

11. Data Model / Schema

ID	Entity	Essential fields / invariants
ENT-001	WalletAccount	ID, chain, address; unique chain/address; EOA only in P0
ENT-002	AuthNonce / Session	purpose, owner, nonce hash, expiry, consumed timestamp
ENT-003	ModelRevision	weights digest, tokenizer/template digests, license record, limits
ENT-004	Worker / CapacitySample	approved endpoint, model revision, slots, telemetry time, cost basis
ENT-005	Quote	owner, body hash, revision, fixed rates, cap, expiry, capacity version
ENT-006	Request / Reservation	quote, nonce, state, amount held, deadline, unique owner/intent
ENT-007	Journal / Posting	immutable balanced entries; asset, integer amount, business-event ID
ENT-008	ChainTransfer / Withdrawal	chain, token, transaction, log index, block hash, finality, state
ENT-009	Receipt / ReceiptBatch	canonical payload, private salt, signature, leaf, proof, root, chain status
ENT-010	OutboxJob / AuditEvent	unique intent, retry state, lease, actor, reason, redacted metadata
ENT-011	ResultArtifact	request ID, encrypted result location, expiry; owner-only retrieval

Amounts are integer token atoms, serialized as decimal strings. Never use floating-point monetary arithmetic. Token decimals come from a verified deployment configuration. Balances derive from the journal; caches must reconcile to it.

12. Backend Requirements

MOD-001 identity verifies login and purpose-specific signatures. MOD-002 pricing binds quotes to exact request bytes and model revision. MOD-003 ledger serializes competing reservations and withdrawals. MOD-004 inference adapter reports definitive completion or uncertainty. MOD-005 chain operations track finality and replacement transactions. MOD-006 receipts canonicalize, sign, batch, and verify. MOD-007 operations enforces access and records actions.

ID	Interface	Contract
API-001	`GET /v1/models`	Public versioned catalog and timestamped prices

ID	Interface	Contract
API-002	`POST /v1/quotes`	Bounded text body, wallet, output cap → signed quote
API-003	`POST /v1/chat/completions`	Quote + wallet authorization + idempotency key → response/receipt or request status
API-004	`GET /v1/requests/:id`	Owner-authenticated polling and result retrieval
API-005	`GET /v1/balance`	Available/reserved/pending in explicit token atoms
API-006	`POST /v1/withdrawals`	Signed same-wallet withdrawal + idempotency key
API-007	`GET /v1/receipts/:id`	Owner-authenticated receipt and inclusion proof
API-008	`/v1/auth/challenge`, `/v1/auth/verify`	One-use wallet login challenge and session
API-009	`/v1/ops/*`	Restricted capacity, reconciliation and pause actions

All money/resource writes require idempotency. Reusing a key with a different body returns 409. Errors use stable codes: `invalid_authorization` 401, `insufficient_balance` 402, `quote_expired` 409, `unsupported_capability` 422, `rate_limited` 429, `capacity_unavailable` 503. An accepted but pending request returns 202 with a status URL; no fabricated completion.

13. Frontend Requirements

Build ModelTable, PriceQuote, WalletStatus, BalanceBreakdown, RequestRunner, RequestDetail, WithdrawalForm, ReceiptVerifier, and OpsHealth panels. Each data view needs loading, empty, stale, unauthorized, retryable failure, and completed states.

Render balances from API-005, never by treating total treasury assets as a user balance. Changing a prompt invalidates its quote. Disable submission on wrong network, expired quote, insufficient available funds, or unsupported model capability. Do not require a browser session for agent inference if the signed authorization is valid.

14. Auth / Payments / Integrations

P0 uses an EOA wallet as identity and payer. Browser sessions are secure HTTP-only cookies; agents sign each call using a dedicated funded wallet. There is no OFFPEAK-issued inference API key. A wallet private key remains a credential and must stay outside prompts and logs.

Recommend **USDG as the candidate settlement asset**, since Robinhood's current token directory lists it; pin and independently verify its address, decimals, transfer behavior, and usable funding route before real-money release. Development uses a clearly labeled mock token. Robinhood token contracts

Use a custodial treasury with 1:1 backing for customer liabilities and a separate operating wallet. P0 withdrawal returns the same token on the same chain; it is not a fiat cash-out guarantee. Charge no OFFPEAK withdrawal fee in the capped pilot; pay gas from the operating budget.

Standard x402 is P1. P0's custom wallet-signed ledger authorization must not be advertised as x402 compatible. Support depends on a concrete scheme/network/token implementation, not EVM compatibility alone. x402 reference implementation

15. Analytics

EVT-001 quote created; EVT-002 authorization accepted; EVT-003 request completed/failed; EVT-004 charge finalized; EVT-005 withdrawal finalized; EVT-006 receipt anchored; EVT-007 capacity unavailable.

Measure repeat funded wallets, successful paid requests, contribution margin, quality pass rate, supplier utilization, price distribution, withdrawal completion time, and receipt publication lag. Keep synthetic/test/operator traffic separate. Chain commitments do not establish unique humans or genuine demand.

16. Security / Privacy

Bind signatures to environment, chain, purpose, wallet, body digest, cap, expiry, and nonce. Authenticate GPU traffic using mTLS and an allowlist; block public worker access. Never accept customer-supplied worker URLs, model code, or shell commands.

P0 retains encrypted response artifacts for 24 hours for safe retries, then deletes them. Prompts are processed transiently and not logged. Billing metadata and receipt proofs retain for 90 days by default; accounting retention must be set for the launch jurisdiction before real funds. Record these as product choices, not zero-retention claims.

Public receipt-registry batches contain roots and counts, never prompts, outputs, customer wallet addresses, or raw per-request receipts. Private salted receipts reduce guessability. Token deposits and withdrawals still publicly expose wallet addresses, amounts, and transaction timing; the backend links wallets to billing activity. This is a pseudonymous design, not an anonymity guarantee. Network access can expose IP addresses to infrastructure operators. The public ledger remains permanent.

17. Performance

Proposed pilot targets, not measured results: control-plane p95 overhead below 150 ms excluding inference and chain activity; receipt publication within 15 minutes under healthy conditions; no more than 120 seconds execution per P0 request; 30-second quote TTL; 30-second capacity freshness limit.

Publish model-specific latency only after a fixed prompt-length, output-length, concurrency and hardware benchmark. Admission control must reject excess demand before charging. A slow chain delays funding and withdrawals, not an already funded model request.

18. SEO

Index the product, catalog, public docs, and status pages. Mark wallet, request, withdrawal, and operator routes private/noindex. Publish comparisons only when the underlying configuration, date, and source are explicit.

19. Accessibility

Target WCAG 2.2 AA: keyboard navigation, visible focus, labeled controls, sufficient contrast, screen-reader status updates, and non-color indicators. Quote countdowns must not force focus changes; users can request a fresh quote.

20. Testing / QA

Require real concurrency tests for accounting; replay and cross-environment signature tests; deterministic metering fixtures; GPU crash/timeout tests; chain duplication and reorganization fixtures; withdrawal replacement tests; independent receipt verification; and a browser journey from funding through withdrawal.

Use mock tokens and synthetic traffic in CI. A real GPU integration test and testnet transfer loop are mandatory before pilot launch. Passing mocks alone is insufficient.

21. Deployment

Deploy the control app and runner as containers with managed PostgreSQL, private object storage, a secret manager, paid RPC, and private GPU networking. Separate development, staging/testnet, and production identities and funds.

Robinhood's current docs list mainnet chain ID 4663 and testnet 46630. Revalidate configuration at deployment. Public RPCs are not the recommended production path. Connection documentation

Deposit availability waits for Ethereum-finalized batch inclusion, not a fixed number of L2 blocks. Display interim status. The adapter must verify what the chosen RPC's finality APIs actually mean. Transaction finality

22. Implementation Phases

The companion build document is the execution contract. Its PH-00-PH-11 packets each include objective, scope, dependencies, artifacts, implementation prompt, verification prompt, pass gate, recovery and handoff. No phase advances until its gate passes.

Order: operating contract → scaffold → identity → ledger → GPU/pricing → deposits → paid calls → withdrawals → receipts → console → operations/hardening → deployment.

23. Milestone Plan

Milestone	Exit evidence	Planning estimate
M1: Feasibility	GPU benchmark, model license, chain/asset verification, provisional cost model	1-2 weeks
M2: Complete testnet loop	Fund → run → charge → verify → withdraw	3-5 additional weeks
M3: Capped paid pilot	Security/operational gates passed, 7 days supervised use	2-3 additional weeks

Estimates assume two experienced engineers, working GPU access, and external specialist reviews when needed. They are not a delivery commitment. A new serving engine or unresolved custody arrangement changes the schedule.

24. Risks

Main risks: usable idle supply is too small; off-peak windows do not match demand; poor batching destroys margin; the model fails buyer quality requirements; token rails or treasury access fail; compromised signing keys permit unauthorized charges; receipt commitments are overstated as execution proof.

Before accepting real balances, identify the operating entity and supported jurisdictions and resolve the applicable custody/payment obligations. The architecture deliberately creates financial liabilities; it is not merely a token-gated website.

25. Open Questions

Q-001: What existing GPU engine, hardware access and benchmarks can the team supply? Q-002: Which initial model passes the workload evaluation and license review? Q-003: Which entity and markets will operate the custodial service? Q-004: Is verified USDG funding and withdrawal practical for pilot buyers? Q-005: Does $OFPK solve a measured coordination problem?

These questions do not block local development. GPU/model answers gate PH-04; asset/finality answers gate PH-05; entity/custody answers gate real-money PH-11. Token economics do not gate P0.

26. Assumption Log

ID	Assumption	Consequence
A-001	Existing GPU technology is not publicly established by this research	Define an adapter; validate before claiming readiness
A-002	Start with one model, region and vetted supply relationship	No public marketplace onboarding in P0
A-003	Wallet-signed prepaid per-call billing satisfies initial agent needs	Direct x402 payments remain P1
A-004	Custodial same-token withdrawals are acceptable for a capped pilot	Disclose custody; require reconciled reserves
A-005	USDG is a feasible candidate, pending live verification	No copied USDC address or assumed token decimals
A-006	Open-model quality is sufficient for narrow text workloads	Benchmark before pricing or customer promises
A-007	$OFPK is optional until utility is demonstrated	No supply, sale, emissions or return promises invented
A-008	A 24-hour encrypted result cache is acceptable	Disclose it and verify deletion

27. Acceptance Criteria

ID	Observable pass condition
AC-001	Catalog shows only approved model revisions with healthy capacity; stale data removes availability
AC-002	Lower utilization reduces the quoted rate under fixed costs; no quote falls below the cost floor; accepted rates never change
AC-003	An agent makes a paid call without an OFFPEAK API key; replay, tampering and wrong-domain signatures fail
AC-004	Duplicate deposit events credit once; concurrent calls and withdrawals cannot spend the same available balance
AC-005	A real GPU returns the pinned model revision; deterministic usage produces the specified charge; unsuccessful requests charge zero
AC-006	An authorized available balance reaches the owner wallet once; pending/replaced transactions cannot double-pay
AC-007	An independent verifier checks signature and Merkle inclusion against a finalized root; a changed field invalidates the proof
AC-008	A new user can complete UF-001-004; empty, error, stale and pending states are visible and keyboard accessible
AC-009	Reconciliation detects a one-atom discrepancy; incident controls stop new risk and preserve financial records
AC-010	Pilot shows positive measured contribution margin and acceptable workload quality before unrestricted growth