Impulse AI: Verifiable Decentralized Infrastructure for AI Training

Litepaper

The Impulse AI Team

Remark *This is a summarized version of our technical whitepaper, which focuses on explaining the fundamental intuition behind our mechanisms. As such, we believe this document is suitable for broader audience. We invite the interested reader to check our whitepaper [53] for a more in-depth, technical presentation. We remark that Sections 1, 2 and 10 of this document are taken from the same corresponding Sections in our whitepaper [53].*

1 Introduction

Over the past few years, fine-tuning large pre-trained language and vision models has become the de facto way for organizations to extract bespoke capabilities from foundation models. Whether adopting a multilingual transformer for legal parlance, teaching a vision model to identify manufacturing defects, or customizing a dialog agent for customer service, fine-tuning domain-specific data reliably boosts accuracy and relevance. In practice, teams spin up GPU clusters on major cloud platforms (e.g., AWS, GCP, Azure), rent dedicated instances from specialized providers, or maintain on-premises hardware orchestrated via Kubernetes or other job-scheduling frameworks. Each training job is configured with hyperparameters, training data paths, and checkpointing policies. Engineering teams (or management) then monitor logs, scale resources up or down, and pay per GPU hour, plus storage and networking overhead.

This model, which is inherently centralized, is effective, but it has four interrelated drawbacks. First, cost and utilization inefficiencies abound: long-running clusters incur idle-time expenses, spot instances can be interrupted unpredictably, and cloud providers' opaque pricing makes it hard to compare alternatives. Furthermore, users are often subjected to a tradeoff between high on-demand pricing and inefficiency for long-term rentals. Second, trust in compute integrity is assumed rather than proven. Tricks like truncated batches (run on a third-party infrastructure) or stale gradients can reduce a model's proper training. Third, a single point of failure and a limited set of vendors constrain availability, geographic diversity, and resilience against outages or regulatory disruptions. Fourth, building a comprehensive, tailor-made machine learning infrastructure is non-trivial, often requiring labor and expertise in high demand but in short supply.

At the same time, the rapid proliferation of open-source compute providers, edge-data centers, and GPU-sharing marketplaces suggests that a more flexible, crowd-sourced approach is possible—if only there were a way to guarantee that distributed providers perform the work they claim. Traditional "spot-market" systems lack strong cryptographic assurances or economic deterrents against fraud. Smart contract systems can automate payments, but without verifiable proof of computation, they merely shift trust to oracles or reputation systems that remain vulnerable to collusion and manipulation.

Impulse addresses these gaps by being *the* one-stop AI marketplace that is supported by decentralized infrastructure, by re-imagining fine-tuning as a fully decentralized, incentive-compatible marketplace with built-in verifiable computing. Rather than negotiating VM instances with a handful of hyperscalers, data scientists, engineers, and users in general submit fine-tuning jobs to the Impulse protocol, specifying their budget, performance objectives, and data-access controls in a smart contract. The protocol then utilizes its Orchestration Layer to match jobs with collateral-backed Compute Providers across a global network, leveraging internal scheduling heuristics and external brokering partners. Once a provider accepts a task, the system enforces verifiable training: only a fraction of gradient updates are redundantly checked by randomly selected verifiers, all of whom stake tokens they stand to lose if they rubber-stamp false results. Detected errors trigger the slashing of malicious actors, while honest participants earn proportional rewards in the native token. This user flow is exemplified in Figure 1.

2 Stakeholder Mapping

Before delving into the mechanics of the protocol, it is essential to understand who will participate, what challenges they face today, and how Impulse AI uniquely serves each group. Our discussion is summarized in Table 1 below.

Compute Providers (CPs) Many operators sit on underutilized GPUs, paying power and maintenance costs without predictable revenue. Impulse AI addresses this by matching CPs with model trainers. In our setting, CPs can provide computational power either as provers (i.e., the actor that computes the main job) or as verifiers. Before joining Impulse, a provider stakes tokens; during execution, a random subset of steps is recomputed by verifiers. Upon successful checks, the provider receives a payout, and in the event of any evidence of malicious behavior, their stake is slashed.

Model Trainers and Fine-tuners Model trainers submit a job specification (dataset, model, hyperparameters) via a unified SDK or API. Then, our orchestration layer routes that job across clouds, edge sites, or prosumer hardware, while at the same time ensuring verifiability of the underlying computation, so trainers need never wonder whether paid-for work ran. By decoupling compute from any single provider, standardizing smart-contract calls, and staking correctness, Impulse delivers flexible, cost-effective, and trustless fine-tuning.

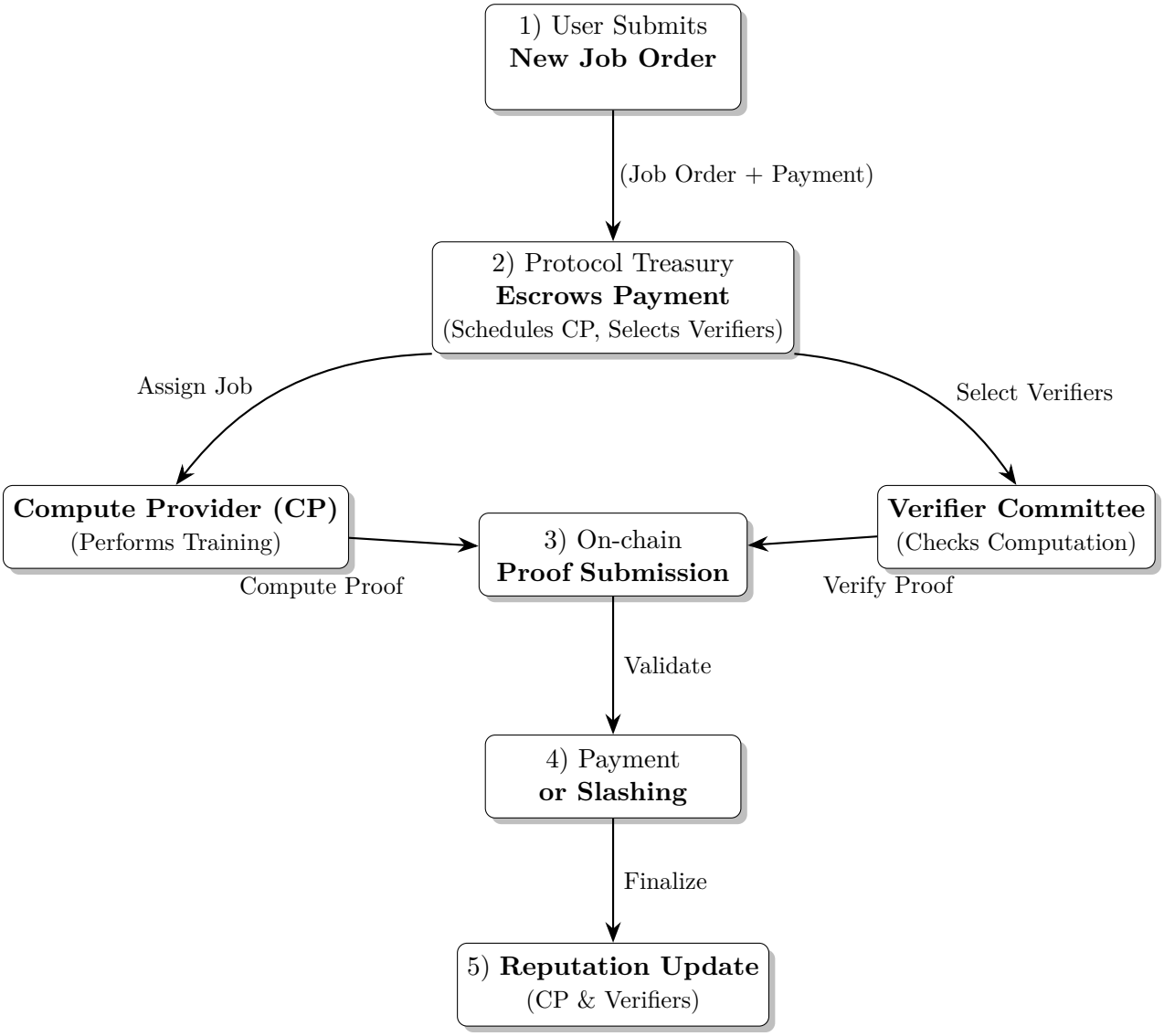

Figure 1: High-level overview of *Impulse*

Retail Contributors Hobbyists and small labs with one or two GPUs often find existing marketplaces inaccessible: minimum commitments are too large, payments too infrequent, and tools too complex. Impulse lowers these barriers by allowing micro-stakes and micro-transactions in native tokens. Anyone with spare GPU cycles can opt in, stake a small amount to signal their commitment and earn proportional rewards for verifiable FLOPs. Over time, retail contributors build on-chain reputations, unlocking larger tasks and higher yields. This democratizes computing, expands total network capacity, and embeds fresh geographic and hardware diversity into the protocol.

Token Holders Native token holders are custodians of the network's long-term success. Impulse AI aligns its incentives by granting governance rights over critical parameters—issuance schedules, slashing thresholds, and KPI targets—and by linking token utility to real-world adoption metrics (compute volume, fee revenue, and reliability). As the protocol scales and usage grows, strong governance participation ensures a healthy network. This creates a flywheel that rewards holders who contribute their voice and expertise to the ecosystem's evolution.

Ecosystem Partners Open-source maintainers, academic researchers, commercial integrators, and independent tool builders each bring unique value to a decentralized compute stack—but today, they navigate siloed services and one-off revenue agreements. Impulse AI provides a standardized SDK and smart-contract interfaces, enabling seamless interoperability between plugin authors, bridge developers, domain-specific orchestration adapters, and the core protocol. High-impact contributions are recognized through Retroactive Public Goods Funding rounds, on-chain bounties, and hackathons, ensuring that maintainers receive token rewards proportional to actual usage, value generated, and community endorsement. By fostering a truly community-driven development model, Impulse transforms partners into co-creators whose work directly shapes the protocol's security, performance, and longevity.

3 On Verifiable Compute

Our goal is to make trust unnecessary by embedding proof-of-correctness and economic accountability directly into the training process. Below we outline the core idea in plain language, and refer the interested reader to our whitepaper [53, Section 3].

Why proofs are needed Faulty gradient step can silently derail a model and waste an entire training budget. Centralized managed services mitigate this risk through reputation and service contracts, but a decentralized marketplace has no such default guardian. We therefore attach a lightweight cryptographic audit to the job itself: for a small, randomly chosen subset of steps, a committee of independent providers recomputes the update. If any discrepancy is found, the errant party's security deposit is automatically slashed and redistributed to honest actors. Because cheaters do not know which steps will be checked, their expected gain from cutting corners is always lower than the expected loss of stake.

Stakeholder	Pain Points	How Impulse AI Helps	Token Benefits
Compute Providers	Idle accelerators, unpredictable revenue, difficult customer acquisition	Stake-backed proof-of-compute, random verifications, per-task payouts, slashing for failures	Transparent rewards, stake-driven governance weight
Model Trainers & Fine-tuners	High costs, having to build ML infra, vendor lock-in, policy constraints	Unified SDK/API, marketplace routing across clouds/edge, cryptographic proof of work	Budget-locked payments, no idle charges, governance votes on pricing & policies
Retail Contributors	High entry barrier, minimal micro-rewards, complex tooling	Micro-stakes, micro-transactions, incremental reputation, verified FLOP accounting	Earn native tokens for small jobs, reputation unlocks larger tasks
Token Holders	Need for sustainable network growth, influence over protocol direction	Governance rights over issuance, slashing, KPI targets; issuance tied to real usage metrics	Voting power, influence on parameter adjustments, alignment with protocol health
Ecosystem Partners	Siloed integrations, ad-hoc revenue, limited funding for public-goods contributions	Standard SDK/contract interfaces, on-chain bounties, RetroPGF grants, hackathons for tool creators	Token grants for contributions, community-weighted funding, seamless liquidity bridges

Table 1: Stakeholder Analysis

Partial redundancy, large savings Naïvely replaying every training step on every verifier would be secure but ruinously expensive. Instead, we sample only a fraction α of steps and assign each of those to a small committee rather than to the whole network. This "partial redundancy'' reduces verification overhead by orders of magnitude while still making detection overwhelmingly likely. The mathematics mirrors random drug-testing in sports: the more you cheat, the higher the odds of being caught at least once. In practice, setting α around one percent already drives the probability of undetected fraud to near zero over the course of a normal epoch.

Stake & Slash Incentives Every prover (the worker who trains) and every verifier posts collateral when joining Impulse. Participants are rewarded upon honest completion, and proven misbehavior burns part or all of the deposit. Because the possible penalty grows with job size while the reward for skipping work is capped by the saved compute cost, rational

actors prefer to do the job correctly. Importantly, honest verifiers also earn a portion of the slashed funds, motivating them to check diligently instead of rubber-stamping results.

From a single prover to many Large language models rarely fit on one GPU. Our framework therefore extends to Distributed Low-Communication (DiLoCo) [13, 12] training, where dozens of workers perform several local updates before synchronising. Verification moves up one level: at each outer synchronisation round we randomly pick a worker and replay its entire local block. This keeps bandwidth low—no extra messages are sent during inner loops—while preserving the same economic deterrents that protect the single-prover case. The mechanism remains robust even if some verifiers collude, provided a strict minority of the committee is dishonest.

Security, liveness and usability The protocol guarantees three properties.

- *Soundness*: Any incorrect update is detected with high probability.
- *Liveness*: Honest Prover and Verifiers can complete all ℓ steps in $O(\ell)$ rounds.
- *Incentive Compatibility*: Rational participants maximize expected utility by behaving honestly.

Looking ahead Section 4 will explain how our orchestration layer assigns jobs to compute providers and verifiers so that the guarantees outlined here scale to thousands of nodes. Subsequent sections describe reputation, slashing, and reward mechanics that tie the whole system together.

4 Orchestration Layer (Job Scheduling)

Here we describe our orchestration layer. The interested reader can find more information on [53, Section 4].

Training jobs submitted to *Impulse* enter a two-stage routing pipeline that pairs user workloads with the most suitable compute providers (CPs) across our globally distributed network. The first stage is an internal scheduler that runs on–chain; the second—still experimental—opens the same optimization problem to third-party "intent" solvers. Together they turn a messy marketplace of heterogeneous GPUs into a coherent, trust-minimizing cloud.

Matching jobs to hardware Every job carries two simple tags: its budget and its expected load (e.g., RAM). Every CP advertises three: free capacity, a live reputation score[1], and the price it is willing to accept. Finding the globally optimal allocation is formally equivalent to a multi-knapsack problem—classic NP-hard territory—so instead of brute force we adopt fast, greedy heuristics that work well in practice.

At the heart of the heuristics is a capacity-degrading score. When a worker wins work its *effective* score is multiplicatively dampened, making it less likely to monopolize subsequent

[1]Reputation is defined in Section 5 and captures historical correctness, availability, and staked collateral.

jobs and nudging future assignments towards under-utilized but reputable peers. A single pass over jobs (sorted by value density) and workers (ordered by residual capacity) produces near-optimal throughput in $O(M \log M + N \log N)$ time, sustaining tens of thousands of jobs and providers per block in simulation. In short, the scheduler behaves like a finely tuned load-balancer that also price-disciplines suppliers through open bidding.

From algorithm to workflow When a user submits a task, the protocol escrows the budget and triggers the internal scheduler. Before joining the network, participating CPs stake collateral as a guarantee of correct behaviour. These participants then receive an execution ticket, and run the verifiable-compute protocol described in Section 3. Upon delivery of proofs the smart contract releases payment; if any step fails, the offender's stake is slashed and the job is automatically re-queued.

Opening the market: external schedulers While the built-in algorithm already outperforms naïve round-robin or flat-price auctions, there is no reason to assume it is globally best. Inspired by DEX "intent markets", future versions will let off-chain solvers post complete allocation matrices, bond collateral, and compete head-to-head. The contract simply picks the proposal that maximizes a public utility function (job value minus reputation penalties) and splits the surplus between network treasury and the winning solver. Invalid or constraint-violating submissions forfeit their bond, keeping the game honest.

Why it matters This layered approach delivers three user-facing benefits. First, it slashes idle time: GPUs with spare cycles are surfaced quickly, while overloaded clusters throttle themselves via the degradation rule. Second, it embeds trust: low-reputation or recently slashed providers are mathematically penalized, so bad actors find it hard to win high-value tasks. Third, it creates a permissionless surface for innovation: anyone—from a traditional cloud broker to a university lab—can plug in an external scheduler and earn fees by routing jobs more cleverly. In combination with verifiable compute, staking, and dynamic reputation, the orchestration layer turns *Impulse* into a self-optimizing, economically secure super-cloud.

5 Reputation

This section is summarized from [53, Section 5].

In a permissionless network any node can show up, claim hash-power and disappear the next day. To keep such fluid membership honest, *Impulse* attaches a living reputation score to every CP. The score is stored on-chain, updates after each completed job, and feeds directly into the scheduler and reward logic introduced earlier. At a glance it measures three things: how often you succeed, how much real work you have done, and how much skin you have in the game. By folding those signals into a single number we create an automatic hierarchy—high-rep nodes are first in line for lucrative tasks, low-rep nodes must build trust before handling mission-critical workloads.

5.1 Three pillars of trust

Success rate is the heartbeat of the metric. After every job the protocol records a performance score r that equals one for flawless execution and slides down towards zero as missed heartbeats, timeouts, or partial failures appear. Instead of averaging all history equally we apply exponential smoothing, so yesterday's behaviour counts more than last year's while still respecting long-term reliability.

Work value acknowledges that finishing a single toy job is easier than sustaining weeks of multi-GPU training. Each budget dollar (or token) a CP earns flows into a running total, again smoothed over time but compressed with a logarithm to account for different job *sizes*. This logarithm also discourages "task spamming": churning through thousands of tiny jobs no longer boosts reputation linearly.

Stake level represents capital at risk. Providers post collateral before joining the network; higher deposits signal confidence in their own uptime and give the network a larger pot to slash if anything goes wrong. Because stake also decays exponentially when unstaked, a node that empties its wallet cannot coast forever on past glories.

Event-driven nudges Routine updates happen after each job, but exceptional events trigger extra adjustments. A single missed heartbeat slightly reduces points off success rate. This reduction is intended to be small enough to matter for close calls in the scheduler yet small enough not to punish rare hiccups. Non-malicious failures (out-of-memory, power blips) cut deeper, while cryptographically proven fraud slashes stake, zeros the immediate performance rating, and shunts the offender to the back of the queue for a cooling-off period. Reputation therefore becomes a live risk indicator that both builders and users can query in real time.

On-chain by design Implementation is deliberately simple: a Solidity mapping from provider address to a tiny struct holding the three smoothed fields plus timestamps. Each update is just a few arithmetic operations—no looping over history, no gas-hungry arrays. Because the score lives on-chain, every other contract (scheduler, staking vault, reward splitter) can consume it trustlessly without external oracles.

Coupling reputation to verifiable work and bonded collateral creates a positive-feedback loop. Honest CPs who finish valuable jobs grow reputation fastest, unlocking bigger assignments and earning higher revenue; dishonesty or chronic unreliability pushes actors into low-budget, high-competition territory until they rebuild trust. The mechanism is Sybil-resistant—each new identity must post fresh stake and climb the ladder from scratch—and resistant to "score farming" because tiny jobs barely move the logarithmic work component. In sum, dynamic reputation forms the social backbone of *Impulse*, translating raw on-chain events into a concise, economically meaningful trust signal.

6 Locking and Slashing

This section is summarized from [53, Section 6]. Collateral is the protocol's lie detector: every actor who touches user data must first lock tokens that can be destroyed if they misbehave. A Prover—the node that runs the actual gradient updates—posts a stake S_{Prover}, while every Verifier escrows S_{Val}. So long as work is completed and proofs check out, the stake simply sits idle; if anything goes wrong, a slice of it is burned and the remainder redistributed to the honest majority. In this way token deposits turn otherwise intangible trust assumptions into hard economic facts.

Two kinds of failure, two penalty scales The system distinguishes intent. Submitting a forged gradient or colluding to rubber-stamp one counts as malicious fraud. Here the slashing fraction is severe (e.g. $\varsigma_{\text{mal}} \approx 75\%$) and the offender is temporarily banned from new work. By design the expected loss—stake multiplied by the probability of detection—always exceeds the small gain from skipping compute, making cheating an irrational gamble.

Hardware crashes, out-of-memory errors or brief network hiccups are treated as non-malicious faults. They still cost money, but the haircut is modest and calibrated so that an honest operator who hits the occasional glitch remains profitable overall. Each fault also nudges the node's live *Success Rate* downward, so chronic instability quietly lowers reputation and pushes the scheduler to favour more reliable peers.

Heartbeat discipline Even between jobs providers must prove they are alive by signing periodic heartbeats. Missing a ping does not slash tokens outright; instead it shaves a fixed percentage off the next reward and, if absenteeism becomes chronic, extends the lock-up window. This lighter touch differentiates flaky connectivity from outright fraud while still encouraging high uptime.

Adaptive lock-ups Freshly deposited collateral cannot be yo-yoed out the moment a job finishes. Every stake is frozen for a minimum period ΔT_{min}, then released linearly over ΔT_{unlock} blocks. Crucially, both intervals shrink as the node's reputation climbs and lengthen if it falls. Long-standing, high-rep providers therefore enjoy superior liquidity, whereas new or recently penalized actors must leave capital on ice until they re-earn trust.

Instant feedback loops Slashing emits an on-chain event that cascades through the system. The reputation contract ingests the exact fraction lost and applies an exponential penalty; the Orchestration Layer immediately re-routes any queued jobs away from nodes whose slashing exceeds a governance-set threshold. As a result, economic, reputational, and operational consequences arrive in seconds, not weeks, and the network heals itself without manual intervention.

Putting numbers on it A representative configuration might burn $\varsigma_{\text{mal}} = 0.75$ of stake for proven fraud, dock only 5% of compute cost for benign failures, fine 1% of rewards per missed heartbeat, and ban malicious nodes for 100,000 blocks. All values are governed

on-chain, allowing the community to tighten or relax parameters as empirical data accumulates. Because these levers directly interact with reputation (Section 5) and scheduling (Section 4), they close the incentive loop that keeps *Impulse* both performant and honest.

7 Rewards Mapping

In this section we provide a short overview of how rewards are distributed in *Impulse*. We follow an approach similar to [59]. This Section is summarized from [53, Section 7].

A fixed schedule mints a number of new tokens every epoch. Governance decides how that pie is carved up: a fraction goes to Compute Providers, another fraction goes to Verifiers, and—when running on a fully decentralized L1—a slice of it goes to to block proposers. The split can evolve over time, giving the DAO a simple throttle for re-balancing security versus throughput.

In addition, every training job pays on-chain fees. A portion of those fees are diverted to the same reward pool. These rewards will later be distributed across CPs. Specifically, within each cohort (providers, verifiers, proposers) the pool is shared proportionally to

$$\text{weight}(i,t) \propto \mathsf{Rep}(i,t) \;\times\; \mathsf{Contribution}(i,t) \;\times\; \mathsf{Stake}(i,t).$$

Here, reputation reflects long-term honesty (Section 5), contribution is normalized proof-of-compute volume for the current epoch, and stake is collateral still locked up after any slashing. Multiplying rather than adding these factors means actors must excel on all three fronts; hoarding stake without doing work, or spamming tiny jobs without stake, barely moves the needle.

Because rewards scale simultaneously with work, stake, and historical quality, the system discourages every common abuse pattern: Sybil attacks become prohibitively expensive, lazy stake-farmers earn little, and reckless over-commitment drags down reputation. Governance can adjust the top-level splits or the mint curve to tighten or loosen monetary policy without touching the core formula, keeping incentives transparent and predictable. In combination with the locking and slashing rules of Section 7, the reward engine creates a flywheel that keeps *Impulse* both performant and honest.

8 Token Economy

This Section is summarized from [53, Section 8].

The native token sits at the heart of *Impulse*. It pays for compute jobs, secures verifiable-compute through bonded collateral, fuels governance, covers gas, and forms the unit in which all rewards are settled. A healthy monetary design must therefore do three things at once: finance core security and verifiability, act as a method of exchange for services, and give governors clear policy levers. We now describe this in more detail.

At the start of each epoch the protocol mints a modest tranche of new tokens. Most of that tranche goes straight to CPs and Verifiers as performance-weighted rewards (see Section 7).

In parallel, every training job pays fees in the same token. A governance-set fraction of those fees rolls back into the reward pot, neatly linking contributor rewards to real usage. Whatever is left over accumulates in a community vault.

Total supply therefore evolves as the sum of four moving parts: new issuance, fee accrual to the vault, fresh collateral locks, and scheduled unlocks. Governance controls two of those levers directly—mint rate and vault allocation—and influences the other two indirectly by adjusting minimum stake and lock-up windows.

Verifiable compute (Section 3) and smart scheduling (Section 4) drive fee flow; dynamic reputation (Section 5) and locking rules (Section 6) gate who can earn those fees; the reward engine (Section 7) redistributes both fees and issuance; and the mechanisms above recycle part of every transaction into either collateral sinks or a community vault. Together, these feedback loops align with long-term protocol health.

9 Minting

We adopt a goal-oriented mechanism as described in [40]. A more detailed version of this section can be found in Section 9 of our whitepaper [53].

In particular, the core idea behind this mechanism is to

1. propose a set of observable, measurable (on-chain), quantities of interest — commonly known as Key Performance Indicators (KPIs), together with some time-dependent target values for these metrics, and
2. Adjust the minting rate according to how close (or far) these network parameters are from their target, in such a way that it creates a positive incentives across the ecosystem.

Naturally, the crux is then to find appropriate KPIs. Rather than tracking dozens of vanity statistics, we select at most **four** core KPIs that drive sustainable growth and resist manipulation:

1. **Verified Compute Volume** Measured as the total number of FLOPs (or GPU-hours) successfully attested via our verifiable-compute protocols each epoch. This directly rewards network utility and is difficult to inflate, since any false claims are caught and slashed.
2. **Active High-Reputation Providers** The count of distinct providers whose on-chain reputation exceeds a threshold (e.g. $\mathsf{Rep}_i > 0.8$) and who completed at least one job in the period. This results in incentivized growth and reliability, making Sybil farming prohibitively costly.
3. **Fee Revenue Growth** The percentage increase in on-chain fee revenue $\Phi(t)$ compared to a moving average. As real users drive demand, higher fees signal healthy adoption, hence, this metric inherently ties net inflation to actual usage.

4. **Job Success Rate** The fraction of submitted jobs that complete all verifiable-compute checks without slashing or timeouts. A high success rate reflects network robustness and a usable user experience; providers cannot game this metric without suffering economic penalties for failed tasks.

10 Usability and Roadmap

Impulse AI is designed not just as a powerful on-chain compute protocol, but as a developer-friendly platform that anyone can adopt in minutes. Today, customers have two integration paths:

1. they can interact directly with the Impulse smart contracts—submitting jobs, staking tokens, and monitoring reputation—or,
2. more commonly, they can use our high-level SDK, RESTful APIs, and web interface.

In practice, most users will invoke the Impulse SDK, which streamlines the entire workflow: uploading datasets, selecting a pre-trained model, tuning hyperparameters, and dispatching a fine-tuning job with a few lines of code.

We believe that exceptional developer tooling is as critical as a robust protocol core. Accordingly, Impulse AI will underwrite and collaborate with third-party teams to build extensions, IDE integrations, and specialized dashboards atop our SDK. A dedicated grants program will fund companies creating value-add tools.

Looking forward, the SDK and UI will grow to support reinforcement-learning workflows, classical deep-learning pipelines, and full pre-training loops in addition to fine-tuning. We will layer in inference capabilities—enabling retrieval-augmented generation, multi-agent decision systems, and real-time chat interfaces—so that one unified SDK covers the entire model lifecycle. Ultimately, we envision an AI assistant embedded in our console —a conversational agent that guides users through data preparation, model selection, deployment, and monitoring, all without requiring handcrafted code.

Beneath this developer surface lies a two-pronged technical roadmap. On the protocol side, we will expand our verifiable compute primitives to include zero-knowledge proofs that hide raw data while guaranteeing step-by-step correctness, support fully heterogeneous distributed training across arbitrary GPU topologies, and implement hardware-bound attestations that prove the exact GPU performed the work. We will integrate Trusted Execution Environments and homomorphic encryption to enable privacy-sensitive and regulatory-compliant training and inference. Mirroring these advances, the inference protocol will gain on-chain proofs of faithful execution, distributed low-latency inference orchestration, and privacy-preserving pipelines for user data.

Simultaneously, our developer toolchain will evolve in lockstep. We will introduce a custom model registry, allowing teams to import and version their private or proprietary architectures alongside open-source foundations. Multi-modal support will enable images, audio, video, and structured data tasks through the same simple API patterns. Deeper integration

with Jupyter notebooks will allow inline proof verification, real-time metric visualization, and interactive debugging. Built-in model evaluation suites will automatically run benchmarks, performance profiles, and fairness audits during both training and inference. Finally, as AI assistants mature, our UI will offer chat-based setup wizards, cost-optimization advisors, and governance proposal drafting aids—making Impulse not only the most rigorous verifiable compute platform, but also the most delightful and productive environment for AI developers.

IMPORTANT NOTICE: DISCLAIMER

This document is for informational purposes only and does not constitute a prospectus, an offer document, an offer of securities, a solicitation for investment, or any offer to sell any product, item, or asset (whether digital or otherwise). Furthermore, this document does not constitute legal, financial, tax, or other professional advice.

The information presented in this document is subject to change or update without notice. The Impulse AI team makes no representations or warranties, express or implied, as to the accuracy or completeness of the information contained herein, and expressly disclaims any and all liability that may be based on such information or errors or omissions thereof.

This document contains forward-looking statements. These forward-looking statements are not guarantees of future performance and are subject to risks, uncertainties, and other factors, some of which are beyond the team's control and could cause actual results to differ materially from those expressed or implied by these forward-looking statements. The Impulse AI protocol is currently in development and its final structure and functionality may differ from what is described in this document.

References

[1] Akash: Gpu pricing and availability. https://akash.network/pricing/gpus/ (2025), online Article. Accessed 03.12.2025

[2] Assran, M., Loizou, N., Ballas, N., Rabbat, M.: Stochastic gradient push for distributed deep learning. In: International Conference on Machine Learning (2019)

[3] Bellachia, A.A., Bouchiha, M.A., Ghamri-Doudane, Y., Rabah, M.: Verifbfl: Leveraging zk-SNARKs for a verifiable blockchained federated learning. arXiv preprint arXiv:2501.04319 (2025)

[4] Bommasani, R., Hudson, D.A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M.S., et al.: On the opportunities and risks of Foundation Models. arXiv preprint arXiv:2108.07258 (2021)

[5] Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., et al.: Language Models Are Few-Shot Learners. Advances in Neural Information Processing Systems (NeurIPS) (2020)

[6] Buterin, V.: Proof of Stake: The Making of Ethereum and the Philosophy of Blockchains. Seven Stories Press (2022)

[7] Chainlink: Chainlink: A decentralized oracle network. https://research.chain.link/whitepaper-v1.pdf (2017), accessed: 2025-02-19

[8] Chainlink Labs: Chainlink 2.0: Next steps in the evolution of decentralized oracle networks. In: Whitepaper (2021), URL https://research.chain.link/whitepaper-v2.pdf

[9] Chen, B.J., Waiwitlikhit, S., Stoica, I., Kang, D.: ZKML: An optimizing system for ML inference in zero-knowledge proofs. In: Proceedings of the Nineteenth European Conference on Computer Systems, pp. 560–574 (2024)

[10] Choi, K., Manoj, A., Bonneau, J.: Sok: Distributed randomness beacons. In: 2023 IEEE Symposium on Security and Privacy (SP), pp. 75–92, IEEE (2023)

[11] Diskin, M., Bukhtiyarov, A., Ryabinin, M., Saulnier, L., Lhoest, Q., Sinitsin, A., Popov, D., Pyrkin, D., Borzunov, A., Wolf, T., Pekhimenko, G., et al.: Distributed deep learning in open collaborations. In: Advances in Neural Information Processing Systems (2021)

[12] Douillard, A., Donchev, Y., Rush, K., Kale, S., Charles, Z., Garrett, Z., Teston, G., Lacey, D., McIlroy, R., Shen, J., et al.: Streaming DiLoCo with overlapping communication: Towards a distributed free lunch. arXiv preprint arXiv:2501.18512 (2025)

[13] Douillard, A., Feng, Q., Rusu, A.A., Chhaparia, R., Donchev, Y., Kuncoro, A., Ranzato, M., Szlam, A., Shen, J.: DiLoCo: Distributed low-communication training of language models. arXiv preprint arXiv:2311.08105 (2023)

[14] Foundation, T.E.: Academic grants round 2025 wishlist (2025), URL https://efdn.notion.site/Academic-Grants-Round-2025-Wishlist-17bd9895554180f9a9c1e98d1eee7aec, accessed: 2025-03-06

[15] Gensyn: Gensyn Whitepaper: Decentralized infrastructure for distributed machine learning. https://gensyn.ai/whitepaper.pdf (2023), accessed: 2025-02-19

[16] Golem Factory GmbH: The Golem project: Crowdfunding a decentralized computing network (2016), URL https://cdn.prod.website-files.com/62446d07873fde065cbcb8d5/62446d07873fdeb626bcb927_Golemwhitepaper.pdf

[17] Google: Vm instance pricing. https://cloud.google.com/compute/vm-instance-pricing?hl=en#section-5 (2025), online Article. Accessed 03.12.2025

[18] Hagerup, T., Rüb, C.: A guided tour of chernoff bounds. Information Processing Letters **33**(6), 305–308 (1990), doi:10.1016/0020-0190(90)90013-F

[19] Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al.: LoRA: Low-rank adaptation of large language models. ICLR **1**(2), 3 (2022)

[20] Huang, Y., Cheng, Y., Bapna, A., Firat, O., Chen, M., Chen, Z., Wu, Y., et al.: Gpipe: Efficient training of giant neural networks using pipeline parallelism. In: Advances in Neural Information Processing Systems (2019)

[21] Jaghouar, S., Ong, J.M., Hagemann, J.: OpenDiLoCo: An open-source framework for globally distributed low-communication training. arXiv preprint arXiv:2407.07852 (2024)

[22] Jia, H., Yaghini, M., Choquette-Choo, C.A., Dullerud, N., Thudi, A., Chandrasekaran, V., Papernot, N.: Proof-of-Learning: Definitions and practice. In: 2021 IEEE Symposium on Security and Privacy (SP), pp. 1039–1056, IEEE (2021)

[23] Jung, S., Kim, H., Kim, Y.: zkCNN: Zero-knowledge proofs for convolutional neural network inference. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (2021)

[24] Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A.N., et al.: Advances and open problems in federated learning. Foundations and Trends in Machine Learning (2021)

[25] Kalodner, H., Goldfeder, S., Chen, X., Weinberg, S.M., Felten, E.W.: Arbitrum: Scalable, private smart contracts. In: 27th USENIX Security Symposium, pp. 1353–1370 (2018)

[26] Keršič, V., Karakatič, S., Turkanović, M.: On-Chain Zero-Knowledge Machine Learning: An overview and comparison. Journal of King Saud University-Computer and Information Sciences p. 102207 (2024)

[27] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[28] Kulkarni, M., Gupta, N., Delworth, S.: Decentralized training of foundation models in heterogeneous environments. arXiv preprint arXiv:2206.01288 (2022)

[29] Lavin, R., Liu, X., Mohanty, H., Norman, L., Zaarour, G., Krishnamachari, B.: A survey on the applications of zero-knowledge proofs. arXiv preprint arXiv:2408.00243 (2024)

[30] LeCun, Y.: The mnist database of handwritten digits. http://yann.lecun.com/exdb/mnist/ (1998)

[31] Li, T., Sahu, A.K., Talwalkar, A., Smith, V.: Federated Learning: Challenges, methods, and future directions. IEEE Signal Processing Magazine (2020)

[32] Lihu, A., Du, J., Barjaktarevic, I., Gerzanics, P., Harvilla, M.: A Proof of Useful Work for Artificial Intelligence on the Blockchain. arXiv preprint arXiv:2001.09244 (2020)

[33] Lin, J., Song, C., He, K., Wang, L., Hopcroft, J.E.: Nesterov accelerated gradient and scale invariance for adversarial attacks. arXiv preprint arXiv:1908.06281 (2019)

[34] Lin, Y., Han, S., Mao, H., Wang, Y., Dally, W.J.: Deep gradient compression: Reducing the communication bandwidth for distributed training. In: International Conference on Learning Representations (2018)

[35] Liu, M., Wong, H.S.P.: The path to a 1-trillion-transistor gpu: Ai's boom demands new chip technology. IEEE Spectrum **61**(7), 22–27 (2024)

[36] Liu, Q., et al.: Proof-of-Useful-Work: A consensus protocol for decentralized machine learning. https://arxiv.org/abs/1909.07962 (2019)

[37] Liu, S.T., Yu, J., Steeves, J.: Poster: Solving the free-rider problem in Bittensor. In: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, pp. 5045–5047 (2024)

[38] Madrigal-Cianci, J.P.: Analysis of filecoin gas: Modelling and uncertainty quantification (2023), URL https://www.youtube.com/watch?v=pRb1BSPZvRk, accessed: 2025-03-06

[39] Madrigal-Cianci, J.P.: Utility and demand for block space (2023), URL https://hackmd.io/LQ8Um2zURFeoGtjoGpiVyA?both=, accessed: 2025-03-06

[40] Madrigal-Cianci, J.P.: Bullish Minting (2025), URL https://hackmd.io/1_Ipeo87R8eE6JO631CniA?view, blog post. Consulted on March 4, 2025

[41] McMahan, H.B., Moore, E., Ramage, D., Hampson, S., Agueray Arcas, B.: Communication-efficient learning of deep networks from decentralized data. In: Proceedings of AISTATS (2017)

[42] Offchain Labs: Arbitrum Rollup: Scalable, private smart contracts. https://developer.offchainlabs.com/docs/intro (2021), accessed: 2025-02-19

[43] Parno, B., Howell, A., Gentry, C., Raykova, M.: Pinocchio: Nearly practical verifiable computation. In: 2013 IEEE Symposium on Security and Privacy, pp. 238–252, IEEE (2013), doi:10.1109/SP.2013.31

[44] Perhats, M., Ferreira, M.X., Cortes-Cubero, A., Karra, K.: Local protocol (2024), URL https://palette-labs-inc.github.io/, accessed: 2025-03-06

[45] Rajbhandari, S., Rasley, J., Ruwase, O., He, Y.: Zero: Memory optimizations toward training trillion parameter models. https://arxiv.org/abs/1910.02054 (2020), arXiv:1910.02054

[46] Rao, Y., Steeves, J., Shaabana, A., Attevelt, D., McAteer, M.: Bittensor: A peer-to-peer intelligence market. arXiv preprint arXiv:2003.03917 (2020)

[47] Sergeev, A., Del Balso, M.: Horovod: Fast and easy distributed deep learning in tensorflow. https://arxiv.org/abs/1802.05799 (2018), arXiv:1802.05799

[48] Shoeybi, M., Patwary, M.P., Puri, R., LeGresley, P., Casper, J., Catanzaro, B.: Megatron-lm: Training multi-billion parameter language models using model parallelism. https://arxiv.org/abs/1909.08053 (2019), arXiv:1909.08053

[49] Sun, H., Li, J., Zhang, H.: zkllm: Zero knowledge proofs for large language models (2024)

[50] Team, T.A.: Aethir whitepaper. https://3028335560-files.gitbook.io/~/files/v0/b/gitbook-x-prod.appspot.com/o/spaces%2FlJdZs7NyMJ6Ewm4U1eRP%2Fuploads%2FOVdpd7QoNIDAZdfpGrGV%2FAethir%20Whitepaper.pdf?alt=media&token=ec38bfde-1668-472d-97d7-a48fb1300703 (2024), online Article. Accessed 03.17.2025

[51] Team, T.G.: Gensyn litepaper. https://docs.gensyn.ai/litepaper (2022), online Article. Accessed 03.17.2025

[52] Team, T.H.: Heurist whitepaper. https://www.heurist.ai/whitepaper (2024), online Article. Accessed 03.17.2025

[53] Team, T.I.A.: Impulse AI: Verifiable Decentralized Infrastructure for AI Training (2025), URL https://github.com/impulse-ai/whitepaper/blob/main/Impulse_AI_Whitepaper.pdf, whitepaper. Consulted on July 22, 2025

[54] Team, T.P.I.: Introducing prime intellect's protocol & testnet: A peer-to-peer compute and intelligence network. https://www.primeintellect.ai/blog/protocol (2024), online Article. Accessed 03.17.2025

[55] Teutsch, J., Reitwießner, C.: A scalable verification solution for blockchains. In: Aspects of Computation and Automata Theory with Applications, pp. 377–424, World Scientific (2024)

[56] Thaler, J., et al.: Proofs, Arguments, and Zero-Knowledge. Foundations and Trends® in Privacy and Security **4**(2–4), 117–660 (2022)

[57] Wang, S., et al.: BlockFL: A blockchain-based federated learning framework. In: IEEE International Conference on Communications (ICC) (2020)

[58] Wikipedia: 2020–2023 global chip shortage (2023), URL https://en.wikipedia.org/wiki/2020%E2%80%932023_global_chip_shortage, accessed: 2025-03-06

[59] Woodhead, P., Cortes-Cubero, A.: Checker network protocol: Litepaper (2025), URL https://blog.checker.network/posts/checker-network-litepaper-1, accessed: 2025-03-06

[60] Woodhead, P., Team, T.C.: Checker network protocol: Economic white paper (2025), URL https://bafkreiaw4vizzgufxse2bohdzsgt57zvvlduajekm4n6ljh6l5hf2tg5iq.ipfs.w3s.link/

[61] Xu, L., Zhang, B., Chen, W.: VERITAS: A framework for verifiable federated learning. Proceedings of the 40th International Symposium on Reliable Distributed Systems (2021)

[62] Zhang, Y., Wang, S.: Proof of sampling: A nash equilibrium-secured verification protocol for decentralized systems. arXiv preprint arXiv:2405.00295 (2024)

[63] Zhao, Z., Fang, Z., Wang, X., Chen, X., Su, H., Xiao, H., Zhou, Y.: Proof-of-Learning with incentive security. arXiv preprint arXiv:2404.09005 (2024)

[64] Zuo, X., Wang, M., Zhu, T., Zhang, L., Ye, D., Yu, S., Zhou, W.: Federated trustchain: Blockchain-enhanced llm training and unlearning. https://arxiv.org/abs/2406.04076 (2024), arXiv:2406.04076