Zansoc: A Web3 P2P Distributed Cloud Computing Platform

The Problem with Traditional Cloud Computing

High Costs: Today's cloud infrastructure is dominated by a few providers (Amazon AWS, Google Cloud, Microsoft Azure) who collectively control ~65-70% of the market [1]. This oligopoly leads to premium pricing. Many businesses find that while cloud is convenient initially, its costs balloon at scale – eating into profit margins. In fact, an analysis by Andreessen Horowitz found that across 50 top software firms, over $100 **billion** in market value was effectively lost due to the *excess cost of cloud* compared to running their own infrastructure [2]. Some companies like Dropbox even saved $75M by "repatriating" infrastructure from the public cloud [3]. Clearly, traditional clouds charge high markups for the convenience and scalability they provide. Smaller startups and individuals often can't afford hefty monthly cloud bills, and even large enterprises see cloud costs as a "trillion-dollar paradox" – paying a premium that eventually outweighs cloud's benefits as they scale [4].

Privacy and Security Risks: Relying on centralized providers means trusting a third party with sensitive data. When data and computing are centralized in one provider's servers, they become an attractive target for breaches and misuse. Past incidents show that misconfigured cloud storage or provider-side attacks can expose customer data. Moreover, a central cloud creates a single point of failure – if the provider has an outage or issue, it can bring down **everyone** relying on it. As one analysis noted, when data is centralized in the cloud it is "highly accessible but also highly vulnerable to security threats, data breaches, and privacy violations" [5]. There is also the issue of data sovereignty and surveillance – data stored on big clouds may be subject to government subpoenas or provider policies beyond the user's control. Users and companies have limited transparency into how their data is handled internally. This loss of control and privacy is a growing concern.

Centralization and Censorship: The dominance of a few cloud companies also raises concerns about centralization of the internet's infrastructure. A decision by one company can have outsized impact on the web. For example, when Amazon's AWS decided to suspend service to a social network (Parler) in 2021, that platform went completely offline, demonstrating the immense power cloud hosts wield over their clients [6]. Centralization means innovation is bottlenecked by the policies and pricing of a few players, and services can be de-platformed if they run afoul of provider rules. Additionally, outages in a single large data center can knock out thousands of applications at once. In summary, the current cloud model suffers from high costs, inherent privacy/security vulnerabilities, and over-centralization that conflicts with the resilient, open ethos of the internet.

Zansoc's Vision: Decentralize the Cloud – Every House a Datacenter

Zansoc envisions a world where **every house is a datacenter and every computer is a node**. Instead of renting compute power from a handful of giant data centers, why not tap into the vast underutilized computing resources in people's homes and offices? Zansoc's mission is to transform the internet's infrastructure from centralized silos into a **distributed, peer-to-peer (P2P) cloud** powered by its users. By leveraging blockchain and P2P technology, Zansoc will enable anyone with a computer to contribute

computing power, storage, and bandwidth to a global network – and get paid for it. This is a paradigm shift from Web2 to **Web3 cloud infrastructure**: users become both providers and consumers, owning the network collectively rather than relying on Big Tech.

At its core, Zansoc aims to **democratize cloud computing**. Just as Airbnb turned spare bedrooms into a global hotel network and Uber turned personal cars into a transportation fleet, Zansoc will turn idle CPUs, GPUs, and hard drives into a community-run "supercloud." This not only cuts costs (by utilizing existing hardware efficiently) but also eliminates centralized control. Data and computations on Zansoc are spread across many nodes, eliminating single points of failure and greatly enhancing privacy – no central company can snoop on or shut down your application. Every participant in Zansoc has a stake and identity on the blockchain, ensuring transparency and trust in the network.

Using Blockchain for Trust: Blockchains enable mutual strangers to coordinate and trust each other's contributions without a central authority. Zansoc uses blockchain smart contracts to track contributions, verify work, and handle payments fairly. This vision means a developer in one country could deploy an application that runs using spare computing capacity from computers across the globe, while dozens of everyday people earn crypto by contributing their machine's uptime to support that app. **In Zansoc's future, cloud computing becomes a decentralized public utility** – like a power grid of compute – rather than the monopoly of a few corporations.

Architecture and Technology Stack

Achieving Zansoc's vision requires integrating a robust stack of decentralized technologies. The architecture is composed of multiple layers handling **compute**, **storage**, **networking**, **orchestration**, **identity/verification**, and **user interface**. Below is an overview of the key components and technologies in each layer of the Zansoc platform:

- **Decentralized Compute (CPU/GPU):** Zansoc leverages existing decentralized computing frameworks like **Golem**, **Ray**, and **Cudos** to distribute processing tasks. **Golem** provides a marketplace protocol for computing power – a network of nodes that supply CPU/GPU resources on-demand [7]. Zansoc can build on Golem's model to let hosts contribute processing power and clients run jobs on remote CPUs/GPUs securely. **Ray**, an open-source distributed execution framework, is used for parallelizing and orchestrating tasks across multiple nodes in a cluster [8]. Ray allows splitting a computational job into many subtasks that run concurrently on different machines, enabling Zansoc to harness **many nodes for one workload** (true distributed computing, not just one node per job). **Cudos** is another decentralized cloud platform that Zansoc integrates for high-performance computing and blockchain connectivity. It enables deploying virtual machines and AI workloads on a global network of providers [9]. By tapping into Cudos, Zansoc nodes can run full VM instances or containers in a trust-minimized way.
  These technologies combined form Zansoc's **compute layer** – a hybrid of peer-to-peer job distribution (Golem/Cudos marketplace) and cluster-style parallel computing (Ray) for intensive tasks.

- **Distributed Storage (Filecoin):** For data storage, Zansoc utilizes **Filecoin**, the leading decentralized storage network. Filecoin turns cloud storage into an algorithmic market, with storage miners offering disk space and earning tokens in exchange [10]. In Zansoc, when clients need to store files or datasets, the data is split and encrypted across multiple hosts' drives via the Filecoin protocol. **Cryptographic proofs** (such as Proof-of-Replication and Proof-of- Spacetime in Filecoin's design) periodically verify that hosts continue to store the data correctly [11]. This ensures reliable, redundant storage without relying on any single data center. By

integrating Filecoin, Zansoc achieves a **distributed file system** where users' data is held by many nodes (enhancing resilience and privacy) and hosts are compensated for provisioning disk space. The Filecoin network's economic incentives and crypto proofs guarantee data persistence and integrity.

- **Orchestration (Kubernetes Scheduler):** Zansoc uses **Kubernetes** for container orchestration across the distributed network. *Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications* [12]. In Zansoc's context, when a client submits a job, it may be packaged as a Docker container or virtual machine image. Kubernetes (and its scheduler) helps assign that workload to available nodes meeting the requirements. It provides a unified control plane to manage workloads running on many disparate machines. The **Kubernetes Scheduler** considers each node's available CPU, memory, GPU, etc., and schedules tasks accordingly, even doing load-balancing across nodes for large distributed jobs. This ensures efficient use of the entire network's resources and abstracts away the complexity of managing processes on thousands of independent machines. Kubernetes' ability to orchestrate computing, networking, and storage across clusters [13] is extended to work in a decentralized environment for Zansoc.

- **Networking & Bandwidth (Helium, Meson Network):** A truly distributed cloud must also handle networking and content delivery in a decentralized way. Zansoc draws inspiration from **Helium** and **Meson Network** to provide peer-to-peer bandwidth and CDN (Content Delivery Network) services. **Helium** is a people-powered wireless network – a global mesh of user- deployed hotspots providing IoT and wireless coverage, rewarded via blockchain. It proved that a community can run nearly **1 million hotspots worldwide** to provide network services [14]. While Helium focuses on IoT/5G connectivity, Zansoc can leverage similar concepts for internet bandwidth sharing. Hosts could share excess internet bandwidth or even host Helium hotspots, contributing to network connectivity in hard-to-reach areas. **Meson Network**, on the other hand, is a decentralized bandwidth trading platform that turns idle server bandwidth into accelerated content delivery for enterprises [15]. By integrating Meson, Zansoc nodes could earn tokens by serving as edge nodes in a P2P CDN – caching and delivering content (web assets, videos, etc.) to nearby users. This *decentralized CDN layer* means applications running on Zansoc can serve data quickly to end-users via the nearest nodes, much like a traditional CDN but without centralized servers. Overall, Helium and Meson components ensure Zansoc not only distributes compute and storage, but also the network traffic, enabling a fully **P2P cloud with community-provided bandwidth**.

- **Identity and Verifiability (Ethereum DID, zkSync, Truebit, Multi-chain):** Trust in a decentralized network requires robust identity and verification mechanisms. Zansoc employs **Decentralized Identifiers (DIDs)** based on Ethereum to establish unique, self-owned identities for each node (host) and user. *Decentralized identifiers are a new type of identifier that enables verifiable, decentralized digital identity* [16]. An Ethereum-DID gives each host node a cryptographic identity (a DID document anchored on Ethereum or another chain) which can hold reputation data, public keys, and attestations about that node. This helps in **trust management** – for example, clients can choose providers with a proven track record, and malicious actors can be identified and excluded via on-chain reputation. For verification of computations, Zansoc plans to incorporate protocols like **Truebit**. Truebit is a verifiable computing layer that allows complex computations to be done off-chain with a fraud-proof mechanism to verify results [17] [18]. In Zansoc, Truebit's interactive verification can be used to **prove that a node performed a computation correctly**: if a result is disputed, a lightweight verification game on-chain can catch and penalize incorrect results, ensuring trustless correctness without re-executing every task on-chain. Additionally, Zansoc leverages modern blockchain scaling solutions – **zkSync** (a

zero-knowledge rollup on Ethereum) and possibly **Polygon** or **Solana** – to handle transactions and smart contracts with low fees and high throughput. zkSync can reduce transaction fees by 99% compared to Ethereum L1 , which is crucial because Zansoc will have many micro-transactions (payments for small computing tasks, storage fees, etc.). By utilizing Layer-2 networks like zkSync or sidechains like Polygon, Zansoc ensures that payments, identity updates, and other on-chain operations are fast and affordable for users. The platform is chain-agnostic and can deploy smart contracts across Ethereum, Polygon, and Solana as needed – for example, Ethereum for security-critical components (identity registry), a Polygon/zkSync for payment channels and task contracts, and even Solana for specialized integrations (Solana's high throughput could be useful for certain parts of the network state). This multi-chain approach provides flexibility and ultra-low cost, all while maintaining verifiability and security through cryptography (e.g., zero-knowledge proofs for transactions).

- **Monitoring and Analytics (Prometheus & Grafana):** To ensure the network runs smoothly, Zansoc uses **Prometheus** and **Grafana** for monitoring the health and performance of distributed nodes. Prometheus is an open-source metrics collection system, and Grafana is a visualization dashboard. Each Zansoc node will run a lightweight Prometheus exporter that collects stats like CPU usage, memory, network latency, job completion times, etc. This data (with user privacy preserved) is aggregated to give a real-time picture of the whole network's status: total available resources, active jobs, success rates, etc. The **Grafana dashboards** allow both the Zansoc team and the community to see key metrics and ensure the network is highly available and performant. For example, if certain nodes are overloaded or a region's network latency is high, that can be detected and addressed (perhaps by recruiting more nodes in that area). Monitoring is crucial in a distributed cloud to match the reliability of traditional providers – it helps quickly pinpoint failures on specific nodes and maintain a high quality of service through proactive management.

- **Frontend and User Interface (React, Next.js, Tailwind CSS):** On the user-facing side, Zansoc is building a modern web application for clients and hosts. The UI is built with **React** and **Next.js** (a React framework for server-side rendering and scalability) to deliver a smooth, responsive user experience. **Tailwind CSS** is used for styling, allowing a clean and consistent design system. The frontend will include dashboards and controls for both types of users: For **clients**, a cloud services dashboard to submit jobs (compute or storage), track progress, and manage payments. For **hosts**, a node management dashboard to monitor their node's contributions (CPU cycles provided, data stored, bandwidth served) and earnings in real-time. The interface will abstract away the complexity of blockchain interactions – for example, handling wallets, digital identities, and payments under the hood – so that even non-crypto-savvy users can participate. The choice of Next.js enables SEO-friendly pages and fast load times, which is useful for public-facing documentation (e.g., a marketplace view of available nodes) as well as the application itself. Overall, the front-end stack ensures that **Zansoc is accessible and easy to use**, turning the powerful but complex backend (blockchain, distributed systems) into a familiar cloud-like experience for end users.

All these components are orchestrated together to form the Zansoc platform. In simplified terms, Zansoc's architecture looks like this:

- Users interact via a **web portal** (Next.js/React) or CLI to request cloud resources or offer their resources.
- The **orchestration layer** (Kubernetes + Zansoc's scheduler logic) matches supply and demand, deploying workloads on the network.

- The **compute layer** (nodes running Golem/Cudos protocols and Ray) executes tasks across the P2P network.
- The **storage layer** (Filecoin/IPFS integration) stores and retrieves data in a distributed fashion.
- The **network layer** (bandwidth sharing via Meson, Helium where applicable) ensures connectivity and fast content delivery.
- The **blockchain layer** (Ethereum/Polygon/Solana smart contracts, zkSync L2) handles identity (DID), trust (Truebit verification), and payments (transactions, token incentives).
- The **monitoring layer** (Prometheus/Grafana) keeps an eye on everything for reliability.
- Under the hood, all interactions are secured by cryptography – from verifying identities and job results to executing payment settlements.

This comprehensive tech stack positions Zansoc as an **integrated Web3 cloud platform**: it combines the strengths of several decentralized technologies to provide a seamless experience comparable to traditional cloud, but with community ownership, privacy, and cost-efficiency.

How Jobs are Distributed, Executed, and Paid For

A core function of Zansoc is to take **compute or storage jobs** from clients and execute them across the distributed network of hosts in a reliable, verifiable way. Below we describe the lifecycle of a task on Zansoc – from distribution to execution, verification, and payment – for both computing tasks and data storage tasks.

Compute Job Lifecycle

When a client submits a computational job (for example, rendering a video, running an AI model inference, or hosting a game server), the process unfolds in several steps:

1. **Client Task Submission:** The client defines the job requirements through the Zansoc interface. This includes the type of task, required resources (CPU cores, memory, GPU, etc.), any software dependencies (provided as a container image or VM image), and a reward/payment offer for completing the job. The job request is then broadcast to the Zansoc network via a **smart contract or off-chain service** that acts as a job marketplace. The client deposits the payment (in Zansoc tokens or stablecoins) into an escrow smart contract at this time, which will hold the funds until completion.

2. **Matching and Distribution:** Once the job is announced, the Zansoc orchestration layer (using the Kubernetes scheduler combined with on-chain signals) matches the job to suitable host nodes. Host providers running the Zansoc node software receive a notification of the pending task if their resources meet the criteria. Depending on the task's nature, **one or multiple hosts can be selected**:

3. *Single-node assignment:* If the job can run on a single machine (e.g., run a VM for 2 hours), Zansoc will pick the best available host (or allow hosts to bid for the task, offering a price). Criteria include the host's hardware, reputation, price rate, and network latency to the client if relevant.
4. *Multi-node distribution:* If the job is parallelizable (e.g., a rendering job that can split frames, or a big data query that can split computations), Zansoc may split the workload among several hosts. For example, using the Ray framework, it can break the job into chunks that run concurrently on 5 different nodes to speed up completion. This is a key differentiator of Zansoc – unlike some decentralized clouds where a job runs on a single provider, Zansoc can **distribute load across many nodes** for faster or higher-capacity computing.

5. The matching can be facilitated by a **smart contract** that programmatically selects N cheapest available nodes that meet the requirements and asks them to commit to the task by staking a small bond. Alternatively, a coordinator node (selected by the network or run by Zansoc's protocol) might handle the matchmaking off-chain for efficiency, then record the chosen workers on-chain.

6. **Execution on Hosts:** The selected host nodes download the necessary code or container image (for example, from IPFS or a registry). Then they execute the compute task in a sandboxed environment (container or VM). Zansoc's software ensures the task is run securely and isolated, so hosts cannot tamper with it without detection. During execution, hosts might report progress periodically (which can be displayed to the client via the dashboard). If the task is interactive (e.g., the client is running a remote development environment), the client can connect to the host's instance through the Zansoc network (with traffic possibly relayed or accelerated by Meson Network for low latency). **Redundancy:** For critical tasks, Zansoc can run the same job on two hosts in parallel (especially if the task is fast to execute) to later compare results – this guards against one host cheating or failing, at the cost of some extra compute, and provides fallback in case one node goes offline mid-task.

7. **Verification of Results:** Once the host(s) declare the task completed, Zansoc verifies the correctness of the computation. This can happen in a few ways:

8. *Checksum or Test Cases:* For certain jobs, the client can provide expected outputs for known inputs or a way to verify the result (like a hash of the correct output if known in advance). The host must provide outputs that match these checks.
9. *Reputation and Redundancy:* If multiple hosts ran the task, the results are compared – if they converge (e.g., 5 nodes report the same answer for a computation), it's very likely correct. If one node disagrees with the others, that node is flagged as faulty or malicious. Majority voting or a consensus mechanism can be used for deterministic tasks.
10. *Truebit-style Challenge:* For arbitrary computations where the result isn't known, Zansoc uses the **Truebit verification protocol**. After a result is submitted, there is a challenge period during which any other node (a "verifier") can dispute the result if they believe it's wrong. A verifier would re-run the computation (or a part of it) and, if finding a discrepancy, trigger an on-chain **interactive verification game** between the original solver and the challenger. The Truebit smart contracts then step through a protocol to pinpoint the exact step of computation that differs, effectively deciding who is correct. The wrong party (the dishonest solver or a false challenger) loses a stake, and the correct party wins it. If no one challenges the result within the time window, it's assumed correct and final. This mechanism incentivizes honesty – a host won't submit a bad result knowing they could lose their stake if challenged. It provides a **trustless guarantee** that even complex off-chain computations are done correctly [18].
11. In many cases, tasks might be straightforward or short enough that a quick re-execution by the network (perhaps by a randomly selected verifier node) is performed automatically as verification. For heavy jobs, Truebit's game is the fall-back to avoid re-computing everything unless a dispute arises.

12. **Proof-of-Work Validation:** In some scenarios (especially for volunteer computing-style tasks), Zansoc could use **proof-of-work certificates** where the host provides a proof (like a cryptographic hash) that is hard to produce without doing the work. An example might be forcing the host to produce a specific type of hash related to the computation's result – if generating that hash inherently required performing the computation, it serves as a proof of work. However, this is more applicable to certain task types; the general solution is the combination of redundancy and Truebit-like challenges.

13. **Payment Settlement:** After successful verification, the escrowed payment is released. The smart contract handling the job now pays out to the host(s) that performed the work. If multiple hosts contributed (parallel or redundant execution), the reward can be split according to the contribution of each or as pre-defined (for example, each host gets a share, or only the fastest N out of M get rewarded in a competitive setup). **Zansoc's commission** is taken at this stage – e.g., a 10% fee from the total payment goes to the Zansoc platform's treasury or revenue pool (the exact percentage can be adjusted, but 10% is an initial model). The remaining amount (90%) goes to the provider(s). Payment is handled by smart contracts on a blockchain network (such as an L2 for low fee). If Zansoc has its own token, the payment might be done in that token; if the client paid in a stablecoin or other currency, the contract would disburse that (or convert to the host's desired currency through an integrated exchange module).

14. If a host had to stake a security deposit for the task, that stake is also returned post-completion (or increased as a reward) if they were honest. Any slashed stakes from cheating nodes could be used to compensate verifiers or be added to the reward pool.
15. The client receives the output of the job (results data, or persistent service running which they can now use). The client can rate or review the performance, which can feed into the host's reputation (and DID profile) for future users to see.

Throughout this process, the roles of blockchain are crucial but happen behind the scenes: smart contracts manage the **posting of the task, locking of funds, assignment of providers, verification games (if needed), and payouts**. Layer-2 networks like zkSync or sidechains are used so that these interactions are fast and incur negligible fees, making even small micro-transactions (e.g., paying $0.10 for a 5-minute compute task) feasible.

Example: Suppose a client needs 100 GPU-hours of computation to render a video. They offer $100 for it. Zansoc finds 10 hosts each with a decent GPU; it splits the rendering into 10 chunks and each host does 10 hours of work in parallel. Within maybe 11 hours (some overhead for splitting/merging), the job is done – significantly faster than 100 hours on one machine. Each host's results are verified (perhaps by one additional host re-rendering a few random frames as spot-check). The smart contract then pays each host $9 (since 10% fee = $10 to Zansoc, leaving $90, split evenly to $9 each). Each host also might have put up a small stake (say $2) which they get back with maybe a bonus from the client's payment. The client receives the full rendered video much faster and at lower cost than a centralized cloud (which might have charged more than $100 for 100 GPU-hours). This illustrates how **distributed execution** benefits both sides: speed and cost for client, earnings for multiple hosts, and a fee for the platform.

Storage Job Lifecycle

For storage and data delivery, the process is slightly different, focusing on redundancy and ongoing service:

1. **Client Storage Request:** A client (which could be an individual or an application) wants to store data on the Zansoc network – e.g., backup 500 GB of files, or host a dataset for a dApp. Through the interface, the client specifies data size, desired redundancy (how many copies across different nodes), duration of storage, and their payment offer (often a price per GB per month). The data itself might be uploaded via the Zansoc client software, which breaks it into encrypted chunks. Each chunk is content-addressed (using a hash, similar to IPFS) so that anyone can verify the chunk's integrity.

2. **Distribution to Storage Hosts:** Zansoc uses the Filecoin protocol (or a similar decentralized storage mechanism) to distribute these chunks to storage providers in the network. Suppose the

client wants 3x redundancy for safety – the system will select at least 3 independent hosts (preferrably in different geographic regions or on different autonomous networks) for each chunk. Hosts with ample disk space will accept the deal by committing to store the data for the agreed period (they may also stake collateral to guarantee they won't drop the data). The selection can also be done via a marketplace: storage nodes might list their price per GB, and Zansoc picks the most cost-effective set that meets the redundancy and reliability criteria. All selected hosts download the data chunks (which are encrypted, so hosts cannot read the client's content – preserving privacy).

3. **Provable Storage and Maintenance:** Once stored, how do we ensure the hosts truly keep the data over time? This is solved by Filecoin's **proof-of-storage** mechanisms. Storage hosts must continuously provide proofs (posted on-chain or to a verification service) that they still hold the data:

4. **Proof-of-Replication (PoRep):** initially, a host proves it has stored a unique copy of the data chunk by producing a proof that can only be generated by actually storing the data in a sealed form.
5. **Proof-of-Spacetime (PoSt):** periodically (say daily or weekly), hosts produce a short cryptographic proof that they still have the data at a certain time. This is done by sampling random portions of the stored data and verifying the host can produce specific cryptographic outputs from it. These proofs are verified on-chain or by Zansoc's protocols. If a host fails to provide a timely proof, the contract can slash their collateral or reduce their reliability score.
6. Zansoc smart contracts or monitoring services thus **audit storage providers** regularly. This entire process is automated; from the client's perspective, their file is just "in the cloud" with multiple backups.

7. If a host goes offline or loses the data, Zansoc detects it via missing proofs and will automatically re-replicate that chunk to a new host (maintaining the agreed redundancy level). The offending host loses some stake (which can be used to pay the new host who takes over storage).

8. **Retrieval and Access:** The client (or anyone they authorize) can retrieve the data by querying the Zansoc network. This works similar to IPFS or a CDN: given the content hash or file ID, the network finds which hosts have the chunks. Using the **Meson Network** or direct P2P connections, it will download from the nearest or fastest hosts. If multiple hosts have the chunk, parallel downloads or CDN-like edge delivery can accelerate access. If the client is using Zansoc to host a website or application data, the network ensures that when users request that content, they receive it from a nearby Zansoc node (fulfilling the role of a content delivery network). Hosts are incentivized to also serve the data when requested because they could receive small additional payments for bandwidth (similar to how Meson rewards bandwidth contributions). In this way, Zansoc's storage layer doubles as a distributed **data delivery network** – frequently accessed data might be cached by additional nodes for quick access, forming a self-scaling CDN.

9. **Payment for Storage Services:** The client's payment for storage is typically recurring (e.g., monthly fees). The initial payment might cover a certain duration upfront, which the smart contract holds and then releases over time to storage hosts as they continue to prove they are storing the files. For example, if a client pays for 6 months of storage, the contract could release the funds in monthly increments to hosts that remain in compliance (with valid proofs). If a host fails and is replaced, the remaining payments switch to the new host. Zansoc again takes a commission (perhaps 10%) of storage fees as revenue. When the term is up, the client can renew the contract by paying again, or choose to retrieve and not continue. The pricing is expected to be **much lower than centralized cloud storage** services, because it harnesses excess capacity.

In decentralized storage networks like Filecoin and others, prices per GB-month have been very low due to competition among many providers. The client benefits from cost savings, redundancy, and not having to trust a single company with their data.

In summary, Zansoc's job handling ensures that compute tasks and storage needs are **distributed across many nodes, executed reliably with verification, and compensated fairly**. By combining cryptographic proofs, economic incentives, and intelligent orchestration, Zansoc achieves a level of trust and performance similar to centralized clouds while leveraging a decentralized network.

Network Roles: Clients and Hosts

Zansoc participants generally take on one of two primary roles (many users will be both at different times):

- **Client (Service Consumer):** This is a user or organization that needs computing resources or storage from the network. Clients submit tasks or storage requests through Zansoc's platform. They provide the specifications and the payment for the service. Clients are essentially the "customers" of the cloud. Their responsibilities include: defining job parameters, depositing payment for the job, and (optionally) providing any necessary verification data or constraints. They also ultimately receive the result – e.g., the output of a computation or access to their stored files. Clients rely on the network's guarantees for security and correctness, and they can specify preferences like higher redundancy or trusted hardware if needed (for an extra cost). In the Zansoc ecosystem, clients could be indie developers deploying an app, researchers needing computation, video creators rendering graphics, companies seeking cost-effective cloud solutions, or even smart contracts (autonomous clients) requesting off-chain compute.

- **Host (Node Provider):** This is an individual or entity that contributes their device's resources to the Zansoc network in exchange for compensation. A host runs the Zansoc node software on their computer, server, or specialized hardware rig. By doing so, they become part of the distributed cloud, offering some combination of CPU, GPU, storage space, and network bandwidth to the network. Hosts advertise their resource availability (how much CPU/GPU, how many GB of storage, network speed, etc.), and the Zansoc scheduler/marketplace assigns them tasks or data to store based on demand. The responsibilities of a host include: executing assigned compute jobs faithfully (within the agreed time and resource limits), storing data reliably and providing proofs of storage, maintaining a good uptime and network connectivity, and following the protocol rules (e.g., updating their software, staking tokens if required, and not acting maliciously). In return, hosts earn payments for the jobs they complete and the data they host. They essentially monetize idle capacity – for example, a gaming PC's GPU that's unused at night can earn money rendering someone's video via Zansoc. Hosts also often have to **stake a certain amount of Zansoc's token or otherwise build up reputation** as a form of trust collateral; this ensures they have something to lose if they cheat or underperform, aligning their incentives with honest service.

These two roles interact through the Zansoc protocol, typically mediated by smart contracts and the orchestration system. A client doesn't usually pick a specific host manually (though in some cases, a client might request a particular node if they have a private arrangement); instead, the network handles the matching. Similarly, a host doesn't choose a specific client – it more generally offers resources and the network assigns tasks as they come.

Zansoc Facilitator Role: In addition to clients and hosts, the Zansoc platform (initial core team or foundation) plays a facilitator role especially in early stages. They develop the software, maintain the blockchain contracts, and provide support. In the long run, Zansoc aims to be as decentralized as possible – governance of the network could even pass to a DAO of token holders, and the matching of clients to hosts is done algorithmically. But initially, the Zansoc team may run some bootstrap nodes (directory services, blockchain oracles, etc.) to ensure the system runs smoothly.

Both clients and hosts benefit from the network's success: a large number of reliable hosts attracts more clients (with diverse tasks), and more client demand means more earning opportunities for hosts. The Zansoc tokenomics (discussed next) are designed to incentivize positive-sum behavior between these roles.

Smart Contracts and Protocol Mechanics (Distribution, Staking, Verification, Payments)

At the heart of Zansoc is a set of **smart contracts** that enforce the rules of the marketplace and maintain trust without centralized control. These on-chain programs handle task assignments, security deposits, proof verification, and payment distribution in a transparent, automated way. Here's how the smart contract layer underpins key processes:

- **Task Listing and Bidding:** When a client submits a job, a **Job Contract** is created on the blockchain (or an existing contract is invoked with a new task event). This contract includes the task metadata (resource requirements, deadline, etc.), the payment offer, and it escrow-holds the client's funds. It essentially serves as an **open call** to hosts. Hosts monitoring the blockchain (or via off-chain network indexers) see the new task and can respond. In some designs, the first qualified host(s) to accept the task (by calling the contract and pledging to do it) get chosen. In others, there could be a brief bidding period where multiple hosts offer to do the job, possibly at different price points or with different performance commitments, and the contract (or an off-chain scheduler) selects the optimal one(s). The task contract will then lock in the chosen host(s) – possibly requiring them to **stake a bond** at this point (by transferring some tokens into the contract as collateral).

- **Staking and Security Deposits:** Staking is used to ensure **skin in the game** for hosts. For example, a host might be required to stake a certain amount of Zansoc's native token when they register as a node or when they accept a job. This stake can be slashed (partially or wholly confiscated) if the host misbehaves – e.g., fails to deliver results, submits a false result that gets caught in verification, or loses a stored file. By risking their own tokens, hosts are financially incentivized to be honest and reliable. The amount of stake can scale with the value of tasks: higher-value or higher-risk computations might require a larger bond. Staking also helps **prevent Sybil attacks** (where someone spins up many fake nodes to win tasks and cause disruption) because acquiring and locking tokens for each fake node becomes expensive. Zansoc's contracts manage these stakes – when a host successfully completes tasks, their stake is untouched (or even earns them a small bonus or reputation boost); if they cheat, the contract can automatically deduct the penalty. Some of the slashed stake could go as a reward to the party that caught the cheat (the verifier or challenger in Truebit), and some could be burned or added to a general insurance pool.

- **Proof of Work/Computation Verification:** As described earlier, Zansoc uses mechanisms like Truebit's verification game to ensure computations are correct. The smart contracts implement this as follows: The job contract does not immediately release payment when a host submits a

result. Instead, it enters a **verification phase**. The result (for example, an output hash or a state commitment from the computation) is posted on-chain by the host. The contract then waits for a short period (maybe a few blocks or a few minutes) allowing any registered verifier to challenge. Verifiers are a special kind of participant (which could be other idle hosts or independent third parties) who watch for tasks and results they suspect might be wrong. To initiate a challenge, a verifier also puts up a stake and provides evidence (depending on the Truebit protocol specifics, this might be initiating an interactive verification session). The contract then manages the resolution: typically, it will interact with a Truebit solver/verifier contract that steps through the disputed computation piece by piece (possibly creating sub-tasks that are executed on-chain in tiny steps to find the divergence). Once resolved, the contract knows who was honest. If the result was valid, the challenger loses their stake (to discourage frivolous challenges) and the host is proven truthful. If the result was wrong, the host's stake is slashed and possibly awarded to the challenger, and the task may be re-assigned or the correct result (from the challenger's computation) is taken as final. This **game-theoretic on-chain protocol** ensures that *don't just trust, verify* is the rule – nothing is assumed correct until it passes either unanimous[20]
agreement or a challenge test. This process is automated by smart contracts and usually will be invisible to the end user unless a dispute actually occurs.

- **Task Distribution via Contracts:** In some implementations, a central contract might store the registry of all available hosts and their stakes. When a task comes in, a **scheduler function** (which could be part of the contract or triggered by it) selects hosts and assigns the task. However, doing complex scheduling on-chain can be costly, so Zansoc will likely use a hybrid: simple commitments on-chain, with actual selection done by off-chain logic that then calls the chain with the result. For instance, there could be a **"TaskManager" contract** that is maintained by a Zansoc DAO or a set of elected oracles – this contract keeps track of nodes' deposits and a queue of open tasks. It might randomly assign tasks to staked nodes (random selection can be fair and resist manipulation, especially if weighted by reputation or capacity). Alternatively, decentralized oracles could run the Ray scheduler off-chain to decide distribution, then feed the assignment into the contract for enforcement (making sure the chosen hosts then follow through or else get slashed). The exact method can evolve with technology (e.g., emerging decentralized scheduler protocols or using threshold cryptography for randomness in selection). Regardless, the contracts ensure **transparency** – clients can see on-chain which node (DID identity) took their task and that the funds are locked until success is verified.

- **Payment and Commission via Smart Contracts:** Once a task is verified complete, the Job contract triggers the payout. It will transfer the appropriate amount of tokens from escrow to the host's account (which could be an address corresponding to their DID identity). At the same time, it will transfer Zansoc's commission to a specific treasury address. For example, if the task was 10 ZSC (Zansoc Tokens) reward, it sends 9 to the host and 1 to the treasury (10%). All this is recorded on-chain, providing an auditable record of transactions – useful for trust and later tokenomics analysis. In the case of multiple hosts or a storage ongoing payment, the contract might stream payments or split among multiple addresses. If the client over-funded an escrow (say the task ended up using fewer resources than anticipated and Zansoc charges by actual usage), the contract could even refund the difference back to the client. This kind of fine-grained billing is possible with smart contracts executing logic on usage metrics reported.

- **Microtransactions and Layer-2:** Because performing many payments on Ethereum mainnet would be expensive, Zansoc's contracts are likely deployed on a Layer-2 network (like zkSync or Arbitrum) or a sidechain (like Polygon). These platforms can handle many transactions per second at a fraction of a cent each. Zansoc might require clients and hosts to have wallets on those networks. The bridging from L2 to mainnet (for those who want to cash out to e.g.

Ethereum or exchange tokens) can be done when necessary. This architecture ensures that even **micropayments** – e.g., paying 0.001 token for a few seconds of CPU time – can be done efficiently. Additionally, payment channels or state channel technology could be used: a host and client might open a channel for a long-running session and exchange many small payments off- chain, only settling the final net on-chain. However, with modern L2s, the simpler approach is just use the L2 for each task's settlement.

- **Governance and Upgradability:** The smart contracts can also encode governance rules, such as how the commission rate is set or how system upgrades happen. There might be a governance token (Zansoc token doubling as governance voting power) where stakeholders can vote on protocol parameters. If so, the contracts would include functions that only execute upon reaching on-chain consensus from the DAO. This ensures the platform can evolve (e.g., adjusting staking amounts, adding support for new types of resources) in a decentralized way. Initially, though, the Zansoc team might control these parameters to react quickly, with a plan to progressively decentralize control to the community as the network matures.

In essence, the smart contract layer is the **trust backbone** of Zansoc. It handles **matchmaking, insurance (staking), truth verification, and fair payment** – all critical to a functioning decentralized marketplace. By using open-source code and the transparency of blockchain, both clients and hosts can trust that the rules are applied evenly and automatically. This significantly reduces the need to trust Zansoc (the company) or any middleman – the **protocol** enforces good behavior. It also simplifies dispute resolution: the code is law, and if there's a disagreement, the blockchain logic resolves it deterministically (or at least provides a framework like Truebit to resolve it).

Economic and Tokenomic Model

Zansoc not only introduces a new technical model for cloud computing, but also a new economic model. The platform has its own **crypto-token economy** and fee structure to incentivize participation and ensure sustainable growth. In this section, we outline how Zansoc makes money, how hosts earn, the pricing advantages, and the role of the Zansoc token.

- **Revenue Model (How Zansoc Makes Money):** Zansoc will operate on a **commission-based business model**. As described, for every transaction in the network (a completed compute task or a period of storage rental), Zansoc takes a small percentage fee. Currently, the plan is around **10% commission** on the transaction value. For example, if a client pays $10 worth of tokens for a job, $1 goes to Zansoc and $9 to the resource providers. This is similar to how Uber or Airbnb take a cut for facilitating the marketplace. Because Zansoc's overhead is low (no need to maintain physical data centers), this 10% can primarily fund development, support, and improvement of the network. In addition to per-task commissions, Zansoc could have other revenue streams: **premium services** (e.g., a client paying for premium support or specialized security audits might incur a fee), or enterprise subscriptions for businesses that want a stable allotment of distributed resources. However, the bulk of revenue is expected from the high volume of micro-transactions across the network. Another potential source is **token appreciation** – if Zansoc holds a reserve of its own tokens (as many platforms do), as the network usage grows the token value might rise, benefiting the treasury. Zansoc might also charge listing fees to node operators who want to be certified or validated by Zansoc for higher trust (kind of like an enhanced listing).

- **Zansoc Token (Utility and Purpose):** The platform will have a native cryptocurrency token (let's call it ZSC for now) that underpins its economy. This token serves multiple purposes:

- *Medium of Exchange:* It is the primary currency used for transactions in the network. Clients purchase ZSC (or receive it via exchanges) to pay for services, and hosts earn rewards in ZSC for providing resources. Using a token allows global, instant payments without traditional banking friction. It also allows programmatic distribution of fees via smart contracts.
- *Staking and Security:* As discussed, hosts likely need to stake ZSC to participate. This locks up some supply, reducing circulating amount, and ties the host's fortunes to the network's success (since they want the token's value to hold or increase).
- *Governance:* ZSC may function as a governance token. Token holders (which include hosts, long- term clients, and possibly investors) could vote on proposals such as changing fees, upgrading protocols, or treasury spending. This aligns control with those economically invested in the project.
- *Incentives and Rewards:* In early stages, Zansoc might use its token to bootstrap the network by issuing **extra rewards (subsidies)**. For example, early hosts might not only earn the client's payment but also a bonus from Zansoc's token reserve to encourage adoption (this is similar to liquidity mining in DeFi – early users earn outsized rewards). Likewise, early clients might get discounted rates if paying in ZSC token, effectively funded by the platform to encourage trial. Over time these incentives taper as organic usage grows.
- *Burn or Buyback Programs:* To create a sustainable token economy, Zansoc could implement a buyback-and-burn mechanism. A portion of the commissions earned (say that 10%) could be used to buy ZSC tokens from the open market and *burn* them (destroy permanently), reducing supply. This would increase token scarcity and potentially its value, which benefits token holders (including hosts). This approach is used by some platforms to share value with participants indirectly.

The tokenomics will be carefully designed to avoid hyperinflation of the token while still rewarding participants. For instance, there might be a fixed supply or a capped supply of ZSC, with an initial allocation for the team, investors, community, and a pool for mining rewards. If the supply is fixed, then the only issuance to hosts is from actual client payments (plus maybe a limited promotional pool). If it's not fixed, a controlled inflation could feed into rewarding hosts. These details will be outlined in Zansoc's token whitepaper section, but the guiding principle is to make ZSC a **true utility token** that is needed to use the network and thus whose value reflects the usage and success of the network.

- **Host Earnings and ROI:** Hosts in the Zansoc network earn money by renting out their hardware's capabilities. One key question for potential hosts (node operators) is: *Is it profitable?* Zansoc's model is designed to make hosting attractive, providing a new source of passive income. For example, consider a host with a moderately powerful PC. Zansoc estimates that an average node meeting the recommended specs could earn around **₹3000 INR per month** (~$40) in profit by contributing to the network (this figure comes from preliminary trials and assumes steady demand) . The actual earnings depend on how much the node is online and what tasks it handles – nodes with a good GPU or a very fast internet connection might earn more by handling specialized tasks (like AI workloads or serving as bandwidth hubs). The ROI (Return on Investment) timeline for a host can be very enticing: if setting up a node (buying some extra RAM, a better cooling, etc.) costs, say, ₹20,000 ($250), and it yields ₹3000/month, the payback period is roughly **6-8 months** after which the hardware has essentially paid for itself and continues generating profit. Even if one includes electricity and maintenance costs, the distributed cloud model often uses *otherwise idle* resources, so the *incremental* cost is low for the host. Additionally, hosts who join early when the supply of nodes is smaller might get a lot of tasks, hence earning more (plus potential early-bird token bonuses). Zansoc will also maintain some *minimum requirements* for nodes to ensure quality (for instance, requiring at least X CPU cores, Y GB RAM, an SSD, and a stable broadband connection). If someone's personal computer doesn't meet these, they could make a **small investment (a few hundred dollars)** to upgrade,

and Zansoc will provide guidance on this – viewing it as setting up a "micro-datacenter at home." This lowers the barrier to entry and ensures consistency across the network. Overall, the host incentive model is about creating a **decentralized gig economy for computing**: anyone can join and earn, with clear profitability and without needing specialized knowledge (beyond setting up the software).

- **Competitive Pricing for Clients:** One of Zansoc's selling points to clients is significantly lower cost compared to traditional clouds. Because it taps into underutilized resources and cuts out large corporate overhead, Zansoc can pass savings to users. As an illustrative comparison, consider cloud GPU rental: AWS or Azure might charge $2-$3 per hour for a GPU instance. Decentralized marketplaces have shown they can be much cheaper – for example, TensorDock, a startup aggregator of GPU servers, advertises prices *80% less than other clouds* (e.g., high-end GPUs at $0.50/hour vs $2.50/hour on AWS). Zansoc can achieve similar or greater cost reductions across compute, storage, and bandwidth. Storage on Filecoin, for instance, can cost orders of magnitude less than Amazon S3 for archival data. By introducing market competition among thousands of individual providers, prices naturally equilibrate to a low level (since many are just monetizing sunk costs of equipment they already own). Therefore, clients can expect **pricing that's 50-90% cheaper** for equivalent services in many cases. Moreover, with the pay- per-use microtransaction model, clients don't need to over-provision or lock into contracts – they pay only for what they actually consume, down to very fine units. This granular billing can further optimize costs compared to the sometimes coarse billing units of traditional cloud (where you often pay for a full hour or full VM even if you need less). [22]

- **Token Utility and Value Dynamics:** The Zansoc token (ZSC) will be freely tradeable on crypto exchanges, allowing market-driven price discovery. As the network grows (more transactions and demand for cloud resources), the demand for ZSC increases because clients need it to pay for services and hosts might stake more of it. If the supply is limited or growing slower than demand, basic economics suggests the token's price will rise, benefiting holders. This dynamic creates an interesting incentive alignment: Hosts, who often will accumulate ZSC from earnings, benefit not just from the immediate payout but also from token appreciation. Clients benefit because a vibrant token economy means they can easily acquire tokens and even hold some as the ecosystem's "fuel." Zansoc itself might perform **buybacks** using a portion of revenue (as mentioned) or distribute some profits as staking rewards to those who lock tokens (encouraging long-term holding). If the token has governance rights, its value also reflects a say in the future of the platform. It's important to design the token such that it isn't purely speculative – it should circulate within the platform. For example, Zansoc could offer discounts to clients who pay in ZSC vs fiat/stablecoin to drive usage. Also, **cross-chain utility**: ZSC could exist as an ERC-20 on Ethereum, an SPL on Solana, etc., bridged between chains, to maximize accessibility. The token's health is critical; mechanisms will be in place to avoid scenarios like oversupply (which could crash value) or hoarding that limits usability. Usually, releasing tokens gradually (vesting for team/investors, mining over years for rewards) helps ensure a stable growth.

- **Monetization Beyond Commission:** In the long run, Zansoc could explore additional monetization like **marketplace services**. For instance, providing a **built-in exchange** where users can swap ZSC to other currencies (taking a tiny exchange fee), or offering **value-added services** like data analytics, consulting for enterprises who want to integrate with Zansoc, etc. However, these are supplementary; the primary model is take a fraction of the enormous volume of cloud computing transactions. Considering the cloud market is hundreds of billions of dollars globally, even capturing a small percentage of workloads on Zansoc can translate to significant revenue.

In summary, the tokenomic design ensures that all parties have aligned incentives: - Clients get low- cost, flexible services. - Hosts earn revenue and see their stake value grow as the network grows. - Zansoc sustains itself through commissions which scale with usage. - The token ties it all together, fueling the ecosystem and representing the value of the decentralized cloud network.

Roadmap and Future Milestones

Zansoc's journey is unfolding in phased milestones, from initial prototype to a full-fledged decentralized cloud platform. Below is the roadmap highlighting past progress and planned future developments:

- **Phase 1: Prototype (Completed)** – *Foundation via CLI:* In this phase, the core concept was proven with a command-line prototype. The team achieved the fundamental pieces of the host-client journey:
- **Node Onboarding:** Implemented a smooth process for individuals to enroll their machines as hosts via a CLI tool. A host can install Zansoc's software, register (obtaining a DID identity and staking if required), and join the network.
- **Network Formation:** Enabled interlinking of nodes into a peer-to-peer network. The prototype established protocols for nodes to discover each other and maintain connections. Essentially, a mesh network was created where nodes can communicate job requests and data.
- **Compute Job Execution:** Demonstrated deploying a virtual machine for a client on a host through CLI commands. In the prototype, a client could request a basic compute instance; the system would pick an active host, start a VM/container on that host, and allow the client to remotely access it. This validated the end-to-end flow: a user's job running on someone else's computer through the decentralized system.

These three steps were successfully combined, resulting in a working proof-of-concept entirely in the command-line interface. While minimalistic, it confirmed that the technology stack (e.g., Kubernetes control via CLI, P2P connections, etc.) works in practice. By the end of Phase 1, Zansoc had an operational "mini-cloud" of distributed nodes that could share resources.

- **Phase 2: Core Platform Development (Q4 2024)** – *Exchange & Blockchain Integration:* Currently underway is the integration of the blockchain components and the economic layer.
- The focus is on inaugurating the **crypto transaction layer** – effectively the "exchange" aspect needed for payments and trust. Smart contracts on a test network are being developed to handle task escrows, payments, and staking logic. In tandem, Zansoc is implementing the Truebit-inspired verification mechanism on testnet.
- **Internal Marketplace:** This phase will produce the first version of the on-chain marketplace where clients can post jobs and hosts can accept them, with all actions recorded on-chain. We call it an "exchange" because it facilitates the exchange of computing for payment using crypto.

- Security audits and iterative testing are heavy in this phase, to iron out any issues in contract logic or game-theoretic incentives before public launch. By the end of Phase 2, Zansoc will have a fully functional backend platform: CLI 2.0 that not only deploys tasks but also handles payments in a decentralized manner.

- **Phase 3: User Interface & Alpha Launch (Q1 2025)** – *UI/UX Rollout:* With the engine in place, the next step is to make Zansoc user-friendly. In this phase, the **web-based dashboard and portal** built with React/Next.js will be released. This includes:

- A **Client Dashboard** with which users can easily submit jobs, track their running tasks, see cost estimates, and manage outputs – all through a graphical interface instead of CLI.

- A **Host Dashboard** where node operators can monitor their node's status (online/offline, resources in use), earnings, and configure what resources to share. It will also guide new hosts through the registration process step-by-step (potentially with automated checks of their system against requirements).
- The UI will have an integrated crypto wallet (or connect to existing wallets like MetaMask) for handling the token payments seamlessly.

- Launching alongside the UI will be an **Alpha release of the Zansoc network** for a broader set of testers beyond the team. Early adopters, perhaps a closed group of community testers or small clients, will be invited to use the platform through the new interface. This is the *Alpha Testnet* stage – real functionality but using test tokens or limited stakes to ensure everything works with people outside the core devs.

- **Phase 4: Beta Network and Token Launch (Mid 2025)** – *Public Beta & Token Generation Event:* After iterating on feedback from the alpha, Zansoc will open up into a public beta. This entails:

- **Mainnet Deployment:** Deploying the smart contracts to the production blockchain network (or Zansoc's own blockchain if applicable). The beta network will handle real economic value.
- **Token Generation Event (TGE):** Zansoc will formally launch its native token (ZSC). This could involve a public sale or distribution to early supporters. Simultaneously, listing the token on exchanges (DEXs like Uniswap and potentially centralized exchanges) will occur so that new users can acquire it. This is the "crypto exchange launch" milestone – making ZSC widely available and establishing its market value.
- **Incentivized Beta:** The beta network will likely include incentive programs – for example, a host recruitment campaign where the first X hosts or first Y months have extra token rewards, and client bounty programs for trying out various use cases. The goal is to rapidly grow usage and gather performance data in real conditions.

- **Scaling Up:** During this phase, Zansoc targets onboarding at least 1000 small clients within six months of launch [23]. Outreach will be done to indie developers, startups, and communities (like blockchain devs, scientific computing volunteers, etc.) to try Zansoc as an alternative to expensive cloud services. On the supply side, Zansoc will also aim to enlist hundreds and then thousands of hosts – possibly partnering with PC enthusiast communities or mining communities to repurpose GPU farms for Zansoc tasks.

- **Phase 5: Full Launch and Growth (Late 2025 onwards)** – *Production Release:* Zansoc transitions from beta to full production. By this stage, the platform should be stable, secure, and user-friendly. Key activities and goals:

- **Marketing & Partnerships:** Aggressively market Zansoc to drive adoption. Form partnerships with blockchain projects (that need decentralized backend), Web3 developers, and even traditional industries looking to cut cloud costs. Highlight case studies from the beta (for example, a startup saved 60% on hosting costs using Zansoc). Also partner with hardware providers or ISPs in various regions to encourage joining the network (perhaps bundle Zansoc node software with certain routers or NAS devices for easy participation).
- **Continuous Improvement:** Based on the real-world usage, optimize the scheduling algorithms, add features like GPU-specific job scheduling (for AI tasks), more fine-grained SLAs (Service Level Agreements) for enterprise clients, etc. Possibly integrate additional blockchain tech as they mature – e.g., if a new zk-proof method allows verifying certain computations more efficiently, incorporate that.

- **Global Expansion:** Ensure the network has a geographically diverse set of nodes. Work on multi-language support in the UI to attract non-English-speaking regions, especially where people have spare hardware and could greatly benefit from extra income (some developing regions might become key supply-side contributors).
- **Decentralized Governance:** If not already in place, initiate the creation of a **Zansoc DAO** where token holders can start proposing and voting on changes, gradually reducing direct control by the founding team and moving to community-driven management.

- **Exchange Listings & Liquidity:** By this time, aim for ZSC token to be listed on major exchanges (Binance, Coinbase, etc.) to improve liquidity and credibility. This will also make it easier for new users to join (they can buy tokens in their local currency to use the service).

- **Phase 6: Ecosystem and Beyond (2026 and future)** – *Web3 Cloud Ecosystem:* Envisioning a few years out, Zansoc aims to be more than a standalone service; it wants to be part of the fabric of Web3 infrastructure:

- Launch a **marketplace for applications** on Zansoc – for instance, templates or images that others can deploy easily (like a decentralized AWS Marketplace).
- Encourage third-party developers to build tools and services on top of Zansoc. Perhaps someone might build a decentralized **"Heroku"** (platform as a service) using Zansoc as the underlying infrastructure, offering even higher-level convenience to deploy apps without thinking about nodes.
- Explore integration with DeFi and NFT: e.g., using NFTs to represent ownership of certain high-powered nodes or to tokenize future capacity of the network (a way to raise funds or allow trading of resource futures).
- **Research and Innovation:** Continue R&D into cutting-edge areas like **proof-of-useful-work** consensus (could Zansoc's computations also secure a blockchain, achieving two goals at once?), or integrating secure hardware enclaves (TEE) to even allow sensitive enterprise workloads to run on untrusted nodes with full confidentiality.
- **Environmental Impact:** Since Zansoc utilizes existing hardware, it is eco-friendlier than building new data centers. Down the line, quantify the energy saved by utilizing idle machines and possibly create a carbon-offset mechanism or marketing around green computing.

Milestones will be regularly updated as the project evolves, but the roadmap above shows a clear path: from a successful CLI prototype, through building a user-friendly platform and launching the crypto- economic systems, to scaling up and becoming a prominent player in the cloud industry. Our conservative goal is that within **6 months of launch**, we have 1000+ clients and a healthy supply of nodes, and within ~2 years, to capture a noticeable share of the cloud market (even a fraction of a percent of the $700B+ cloud market is significant [24]).

Market Opportunity and Competitive Landscape

A Trillion-Dollar Opportunity

The **cloud computing market** is massive and growing rapidly. By 2025, global cloud spending (including IaaS, PaaS, and SaaS) is projected to reach **$723 billion** for public cloud services alone [24]. Some broader estimates put the overall cloud market (including private and hybrid cloud) at close to $1 trillion by 2025 [25]. This illustrates the sheer scale of what Zansoc is addressing. Even more significant is that nearly **half of IT workloads** in the world have already migrated to cloud and that share is increasing [26]. However, as discussed, current cloud providers have inefficiencies and high margins –

meaning there is a huge pool of value that could be redistributed to users and providers if a more efficient model (like Zansoc's decentralized approach) takes hold.

Moreover, the market is currently dominated by a few players, which, while highly capable, collectively leave certain customer needs unmet (due to their pricing and structure). The traditional cloud providers captured ~68% of the market as of 2024 [27], signifying a *centralized hold*. Zansoc sees an opportunity analogous to how **ride-sharing disrupted taxis** or **Bitcoin disrupted centralized finance** – by leveraging networked technology to unlock value and give users more choices. If Zansoc can even capture a small portion of the workloads – for example, servicing the needs of cost-sensitive startups, open-source communities, or regions under-served by big cloud data centers – it could scale to a multi-billion-dollar platform. And because Web3 users are increasingly looking for decentralized alternatives (for reasons of sovereignty, censorship-resistance, and community ownership), Zansoc fits into a rising trend of **Decentralized Physical Infrastructure Networks (DePIN)**, where real-world infrastructure is coordinated by blockchain tokens [28].

Additionally, emerging technologies like AI, machine learning, IoT, and edge computing are driving **explosive demand for compute and storage**. Centralized clouds struggle to keep up with specialized demands (like GPUs for AI) which is why their prices remain high. Zansoc can agilely meet these trends by onboarding specialized hardware from whoever has it (for instance, many crypto miners have GPUs that could be repurposed for AI tasks when not mining). The timing is right for a decentralized cloud: the general public is more aware of crypto and Web3, hardware is broadly distributed, and software frameworks (like those Zansoc uses) have matured to enable this complex coordination.

In summary, the market opportunity for Zansoc is twofold: **capture some of the existing cloud market** by offering a better value proposition, and **expand the market** by enabling new use cases (like micro-services in regions without data centers, or individuals buying compute by the minute with crypto in a trustless way, etc.) that weren't feasible before. Given the size of the market, even 1% market share would be >$5 billion in annual revenues – an ambitious but not impossible target if the decentralized model proves out.

Competitor Analysis

Zansoc is not alone in recognizing the need for decentralized cloud infrastructure. A few projects and companies are exploring related ideas. We analyze some notable ones:

- **Aleph.im:** Aleph.im is a decentralized cloud platform focusing on providing **serverless computing and decentralized databases** for Web3 applications [29]. It allows developers to offload computation from Ethereum smart contracts to off-chain nodes (Compute Resource Nodes) that run the code and return the results. Aleph.im's strengths are in simplicity (it offers a straightforward API for devs) and its integration with multiple blockchains. However, in Aleph's current model, each offchain computation typically runs on a single node or a predefined set of nodes. **It is decentralized (many independent nodes) but not fully distributed** in the sense of parallelizing single tasks across multiple nodes. There isn't a dynamic multi-node load-sharing for one job; you trust one Aleph node (or a redundant pair) to execute your task. Zansoc's advantage here is the ability to harness **multiple nodes for one client seamlessly**, enabling higher performance for big tasks. Additionally, Aleph.im primarily targets blockchain dev use- cases (like managing off-chain data for dApps), whereas Zansoc targets a broader cloud market (including traditional computing tasks).

- **TensorDock:** TensorDock is a marketplace for GPU computing, advertising cheaper cloud GPUs by aggregating providers [21]. Essentially, it connects users needing GPU instances (for AI, rendering, etc.) with suppliers who have idle GPU servers. TensorDock is more of a Web2 service with a centralized website and payment system, though it taps into distributed resources. The cost savings they claim (up to 80% less than AWS) validate the concept that many small providers can undercut big cloud [22]. The limitation of TensorDock is that it's still a **centralized intermediary** – users have to trust TensorDock to broker the deal and ensure providers deliver. Also, each server rented is a standalone resource; if your job needs multiple servers, you the user must manually orchestrate that (TensorDock doesn't automatically merge them). In contrast, Zansoc's fully decentralized, blockchain-mediated approach removes the central intermediary risk (trust is handled by protocol). Zansoc can also **automatically combine multiple machines** for a single workload when needed, which TensorDock doesn't inherently do (it's focused on single VM provisioning). Nonetheless, TensorDock and similar GPU marketplaces (like Vast.ai, RunPod) are indirect competitors addressing the high-cost GPU niche. Zansoc will compete by automation (no need to manually find/SSH into machines), broader resource offerings (CPU, storage, etc., not just GPU), and trustlessness (smart contracts ensure payment/ result, rather than a company middleman).

- **Akash Network:** Akash is a decentralized cloud compute marketplace built on the Cosmos ecosystem. It allows developers to deploy Docker containers on provider servers in a permissionless way. It's perhaps the closest in spirit to Zansoc in that it aims to be a decentralized alternative to AWS EC2 and similar. Akash providers set prices for their resources and developers bid for those resources through Akash's blockchain auctions. Akash has seen some adoption for hosting blockchain nodes and simple applications. However, one differentiation is that Akash typically provisions entire containers/VMs on a single provider node – it doesn't yet aggregate multiple providers for one deployment (your container runs on one host). Zansoc's Ray and Kubernetes approach could outperform by distributing workloads. Also, Zansoc plans to incorporate **storage and bandwidth** natively (Akash is mostly compute and relies on external solutions for storage like urging users to use IPFS). In short, Zansoc envisions a more comprehensive platform. That said, Akash is a known project; Zansoc will monitor and perhaps even integrate (e.g., use Akash as a fallback or additional resource pool). Our advantage will be in user experience (Akash currently requires some devops knowledge), multi-node orchestration, and the economic model (Zansoc's commission vs Akash's auction model – we believe our model might yield more stable pricing and better host incentives).

- **Filecoin, Storj, Sia (for storage):** On the storage side, Zansoc's competitors are the likes of Filecoin (which we actually incorporate as a component), Storj, Sia, etc., which are decentralized storage networks. These have matured in providing reliable distributed storage at low cost. Zansoc differentiates by being a *unified solution* – whereas those networks handle storage only, Zansoc offers compute+storage+bw together. For many clients, the integration is valuable (e.g., run a computation that directly reads/writes data from the distributed storage, all in one platform). In terms of competition, Zansoc isn't trying to "beat" Filecoin – instead, likely to ride on its success by using it. The market opportunity for storage is huge (data stored globally is in zettabytes), so even leveraging a fraction via Filecoin integration means Zansoc can attract users who need one-stop-shop for cloud.

- **Cudos:** Cudos, which we mentioned in tech stack, is both a collaborator inspiration and a competitor. Cudos provides a blockchain-based cloud where people can deploy VMs and even smart contracts on a distributed network [9]. It positions itself as a *"decentralized computing layer for all chains"*. One could imagine Cudos and Zansoc competing for node providers and tasks. However, Zansoc can differentiate by focusing on the **P2P cluster angle** (Cudos is more

analogous to a decentralized AWS EC2, one VM = one host). Zansoc might also differentiate in the **community aspect** – making it easy for an average person to join (Cudos might require more technical setup and seems also focused on enterprise AI workloads). Nonetheless, Zansoc will keep an eye on Cudos and possibly ensure compatibility (maybe a Zansoc job could run on a Cudos provider if needed, effectively interoperating).

- **Traditional Cloud (AWS, etc.):** Ultimately, the biggest "competitor" is the status quo – the Amazons and Googles. While Zansoc doesn't directly compete head-to-head in the early stage (no single startup can outspend or out-feature AWS initially), it offers a fundamentally different value prop. For certain segments, Zansoc will be far more appealing (on cost, on decentralization, on novel capabilities). Traditional providers are starting to acknowledge decentralization – for example, AWS Outposts and edge computing initiatives try to extend their reach to on-premise locations, but they are still centrally managed. Zansoc flips that model to a bottom-up network. In the long run, if Zansoc and others prove the model, we could see the big providers attempt competing services or even hybrid integrations (for instance, AWS could someday host a marketplace of third-party-owned servers – essentially co-opting decentralization). Zansoc's strategy is to move faster and establish a network effect (lots of nodes and users) that would be hard to replicate by a centralized company without undermining their own business model.

To summarize the competitive landscape: **no one else currently offers the exact combination of features that Zansoc does** – a *fully* distributed cloud where a single application can harness multiple independent nodes across compute, storage, and networking, all coordinated by blockchain. Some competitors are decentralized but not focusing on distribution (Aleph, Akash), others are distributed resources but not trustless (TensorDock, vast.ai), and others cover only a slice (Filecoin for storage). This means Zansoc has a strong differentiation as a *truly holistic Web3 cloud service*. The challenge will be execution – making the UX smooth and attracting enough supply/demand to achieve critical mass. But once achieved, **network effects** kick in (more nodes → better service → more clients → better rewards
* attracts more nodes, and so on), giving Zansoc a defensible position.

Zansoc's Key Advantages

- **Truly Distributed Workloads:** Unlike many "decentralized" compute solutions that ultimately run each job on one node, Zansoc can split and load-balance tasks across many nodes. This means higher performance for large-scale jobs and no single node becomes a bottleneck. Effectively, Zansoc can act as a **distributed supercomputer** when needed, by harnessing the power of many small machines in parallel.

- **Integrated Services (Compute + Storage + Network):** Zansoc offers a one-stop platform. A decentralized app developer can use Zansoc to run their back-end *and* store their user files *and* serve content to their users – all via one unified interface. This synergy is more convenient than piecing together one service for compute (e.g., Golem), another for storage (Filecoin), another for CDN (maybe no direct analog aside from centralized CDNs). It also allows optimizations – for instance, Zansoc can place computation on nodes that are near where the data is stored to minimize latency (data locality optimization), which separate services couldn't easily coordinate.

- **Cost and Efficiency:** As outlined, Zansoc is extremely cost-competitive. By leveraging idle resources (which are essentially sunk cost for the owners), we can price services at marginal cost, which is far below the premium of cloud giants. This opens the door to users who previously couldn't afford certain levels of compute or who want to do things at a scale that was cost-

prohibitive. Also, hosts earn incremental income, which encourages more to join – a positive feedback loop of efficiency.

- **Community and Ownership:** Zansoc is not just a service, it's an ecosystem that *users can own a part of* through the token. This aligns interests – people are more likely to evangelize and improve a platform when they have a stake in it. Over time, Zansoc could become self-governed by its community, making it resilient and adaptive. In contrast, using AWS gives users no ownership; they are simply customers. Some Web3 projects will prefer to "build on something that they have a voice in" – Zansoc offers that alignment.

- **Censorship Resistance and Resilience:** Because Zansoc's infrastructure is decentralized globally, an application running on it is much harder to take down or censor. There is no central kill-switch. This is crucial for applications that value freedom of information or need to operate across borders without dependency on any single country's infrastructure. Even from a reliability standpoint, Zansoc can be more resilient to failures: if one region's nodes go down (say due to a power grid issue), the workload can be shifted to other regions seamlessly. Traditional clouds do have multi-region setups, but they are still limited by the provider's data center footprint. Zansoc's footprint is potentially *everywhere someone has a computer and internet* – which is to say, almost everywhere on the planet.

- **Rapid Scaling and Elasticity:** Need 10,000 servers for a one-hour job? On Zansoc, if the nodes are available, you could theoretically tap into all of them at once without needing any one entity to have that capacity. It can scale in a flash, since it simply involves recruiting more participants. Traditional cloud scaling is limited by how quickly the provider can rack new servers (which is why spot pricing on AWS fluctuates – capacity isn't infinite). Zansoc leverages *latent capacity* that's already out there. If properly harnessed, the upper bound of Zansoc's compute power could far exceed any single provider's fleet, because it aggregates the *entire world's* computing resources.

- **Technical and Business Model Innovation:** Finally, Zansoc's advantage is being a first-mover (or early mover) in combining all these elements. The complexity of building such a system is a high barrier to entry. If Zansoc executes well, by the time others try to replicate it, we would have the network effect and a refined technology. The team's mix of expertise in blockchain, cloud, and distributed systems is relatively unique – it's not just a blockchain project or just a cloud startup, but a fusion. This is an advantage in itself in a field where many competitors come from only one side and might miss pieces of the puzzle.

Conclusion: The Future of Web3 Cloud with Zansoc

The internet is on the cusp of a new era. We are moving from the age of monolithic cloud providers to an era of **decentralized, democratized infrastructure**. In this future, the cloud will be powered by the people, just as cryptocurrencies are powering a people-driven financial system. **Zansoc** stands at the forefront of this transformation, aiming to redefine how we think about computing resources in the same way Bitcoin redefined money or Uber redefined transport.

Imagine a world where launching a tech startup doesn't require a huge server bill – instead, you tap into a global cooperative of machines where costs are low and everyone helping you run your app is also supporting the network because they earn value from it. Imagine scientific researchers being able to run complex simulations not on a restricted grant-based cluster, but by drawing compute from thousands of willing participants around the globe, speeding up discoveries. Envision users in developing countries being able to monetize an old laptop by contributing to the cloud, while local

developers in those regions use the cloud without needing any Silicon Valley company's permission or pricing.

This is the world Zansoc is building: **an open, accessible, and equitable cloud infrastructure** that aligns with Web3 principles of decentralization, privacy, and user empowerment. By turning every house into a datacenter and every computer into a node, we remove the limits imposed by centralized entities. We also cultivate a more **inclusive internet economy** – one where value is shared by many (the node operators, the token holders, the community) instead of concentrated in a few corporate balance sheets.

The road ahead is certainly challenging. Building a distributed cloud means solving cutting-edge problems and ensuring performance and security meet the high bar set by traditional providers. But the progress so far – a working prototype, clear technology blueprint, and strong economic incentives – gives confidence that Zansoc can achieve this ambitious vision. The timing is right, as awareness of decentralization grows and the shortcomings of current clouds become more evident.

In the coming years, as Zansoc grows, it could become **as critical to Web3 as AWS was to Web2**, but with a fundamentally different model. We see Zansoc as a cornerstone of the **next-generation internet** – an internet that is not just connected, but also **collaborative and distributed at its core**.

The cloud of tomorrow will not be a place or a company; it will be a protocol and a community. In that future, **Zansoc** will be known as a pioneer that helped usher in the decentralized cloud revolution, ensuring the internet's infrastructure becomes as free, open, and resilient as the internet's original spirit intended. With every new node that comes online and every new client that deploys on Zansoc, we move one step closer to that brighter, democratized future.

Zansoc invites you to join this journey – together, we will build the cloud **by the people, for the people**, and redefine the very fabric of the digital world. The future of web infrastructure is here, and it is decentralized. Welcome to Zansoc's cloud, where everyone has a part to play in powering the Internet of tomorrow.

---

1 27 **AWS, Microsoft, Google Fight For $90B Q4 2024 Cloud Market Share**
https://www.crn.com/news/cloud/2025/aws-microsoft-google-fight-for-90b-q4-2024-cloud-market-share

2 3 4 **The Cost of Cloud, a Trillion Dollar Paradox | Andreessen Horowitz**
https://a16z.com/the-cost-of-cloud-a-trillion-dollar-paradox/

5 **Data Privacy and Security Concerns for Multi-Cloud Organizations - DATAVERSITY**
https://www.dataversity.net/data-privacy-and-security-concerns-for-multi-cloud-organizations/

6 **How Parler deplatforming shows power of Amazon, cloud providers**
https://www.cnbc.com/2021/01/16/how-parler-deplatforming-shows-power-of-cloud-providers.html

7 **Golem Network**
https://golem.network/

8 **Overview — Ray 2.46.0**
https://docs.ray.io/en/latest/ray-overview/index.html

9 **What is Cudos (CUDOS) - A Comprehensive Overview**
https://www.imperator.co/resources/blog/what-is-cudos

10 **Filecoin: A Decentralized Storage Network | Protocol Labs Research**
https://research.protocol.ai/publications/filecoin-a-decentralized-storage-network/

11 **What is Filecoin**
https://docs.filecoin.io/basics/what-is-filecoin

12 13 **Kubernetes**
https://www.ibm.com/docs/en/blockchain-platform/2.5.3?topic=reference-kubernetes

14 **Protocol Report 2023 Q1 • Helium Foundation**
https://www.helium.foundation/protocol-report-2023-q1

15 **Meson Network Price, msn to USD, Research, News & Fundraising**
https://messari.io/project/meson-network

16 **Decentralized Identifiers (DIDs) v1.0 - W3C**
https://www.w3.org/TR/did-core/

17 18 **Overview | Truebit on Ethereum v1**
https://docs.truebit.io/v1docs

19 **Why zkSync Layer 2 solves Ethereum's transaction fees? - Cruxpool**
https://cruxpool.com/blog/why-is-zksync-a-good-layer-2-solution-to-ethereum-transaction-fees/

20 **Truebit price TRU #4185 - CoinMarketCap**
https://coinmarketcap.com/currencies/truebit/

21 23 **Zansoc-Research.pdf**
file://file-8srCMreBJDbNimSKfkwK76

22 **TensorDock — Easy & Affordable Cloud GPUs**
https://tensordock.com/

24 25 **2025 Cloud Computing Market Size And Trends**
https://www.cloudzero.com/blog/cloud-computing-market-size/

26 **Everything to Know About Cloud Data Privacy | Flexential**
https://www.flexential.com/resources/blog/cloud-data-privacy

28 **Decentralized Infrastructure: Helium - Crypto Council for Innovation**
https://cryptoforinnovation.org/decentralized-infrastructure-helium/

29 **Aleph.im (ALEPH): A Comprehensive Overview of a Decentralized ...**
https://oakresearch.io/en/reports/protocols/aleph-im-comprehensive-overview