Nakshatra — Thesis & Vision

A short statement of what we are building, why, and what we believe. The longer technical report lives in `paper-draft.md`; the architectural contract in `petals-architecture.md`; the full vision in `north-star.md`. This document is the one you read first.

The thesis

A person's intelligence should run on hardware they own.

The cloud has spent fifteen years teaching us that artificial intelligence is a tenancy — that the only real way to use a large model is to rent time on someone else's machine, on someone else's terms, at someone else's price. This is a fact about today's economics, not a fact about the world. The hardware to run a 70-billion-parameter language model already exists in tens of millions of homes — fragmented across a laptop, a desktop, an iMac, a gaming PC, a Raspberry Pi. The software to *pool* that hardware into one coherent inference engine does not yet exist in a serious form.

Nakshatra is our attempt at it.

Why this is possible now

Three things converged inside a single year and made the problem tractable for the first time.

- **Open-weight models caught up to the frontier.** Llama-3, Qwen-2, DeepSeek, Mistral — within a generation of closed-model quality on the workloads that matter for personal use.
- **Consumer GPUs grew up.** A 32 GB RTX 5090, a 24 GB Mac Studio, a $500 AMD card with 16 GB — every household with one tech-enthusiast now owns compute that would have cost a startup six figures five years ago.
- **The home network stopped being a problem.** Tailscale and similar overlays made point-to-point connections across the consumer internet boring. The old "but NAT traversal!" objection to peer-to-peer architectures is dead.

What's missing is software that uses any of this. Nakshatra fills that gap.

What we are building

Three projects. One shared mission. Each small enough to ship on its own; together they compose into something none of them could be alone.

Nakshatra (L2) — the road. An inference engine that splits one large model across many mismatched machines. Each machine holds part of the model's layers; only a small activation vector flows between

them per token. Built on `llama.cpp`, so it runs on every vendor's hardware out of one codebase — CUDA, ROCm, Metal, Vulkan, CPU. v0.1 is alive on a five-machine cross-vendor cluster.

Sthambha (L3) — the pillar. A coordination layer above Nakshatra. Holds the peer registry, the layer cache, the identity primitives. Crucially, it is *also* the soul-persistence layer — Shamir-split identity shards and state snapshots that survive every compute node dying. A pillar on a Raspberry Pi can be the difference between an agent that exists across hardware generations and one that disappears with a broken laptop.

Prithvi (L4) — the being. A long-running personal agent. The cars on the road. Voice, mind, gateway, contracts — the surface a person actually talks to. Owns nothing about inference or coordination; consumes both from below. Portable across laptops, phones, IDEs.

The road, the pillar, the being. Three concerns, three projects, one substrate.

What we believe

1. **Sovereignty is a feature, not a luxury.** A person's AI should not be a tenant of someone else's product. The intuition that produced "host your own server" in 2005 and "self-custody your keys" in 2015 produces "run your own intelligence" in 2026.
2. **Heterogeneity is the default, not the edge case.** A homogeneous fleet of identical GPUs is what a datacenter has, not what a person has. Software that assumes matching hardware is software for somebody else.
3. **Continuity outlives hardware.** Devices die. Agents that live on a single device die with it. The substrate must hold what the compute cannot — identity, memory, the trajectory of a long conversation.
4. **The protocol is the contract; the architecture is the discipline.** Three projects exist instead of one because three projects can't accidentally merge concerns. Each layer's job is small enough to debug. The discipline of the split is what makes the composition work.
5. **The substrate, not the product.** Open weights, open code, open protocols. We are building infrastructure — not a hosted service, not a token launch, not a walled garden. The economics that sustain the work are downstream of the protocol being correct. We earn the next stage by shipping the current one.

Where this goes

If Nakshatra ships and grows past a private alpha, what becomes possible looks like this:

- A friend's old gaming PC, a sibling's MacBook, a parent's iMac, your own Pi — all running together as one cluster. The cost of adding a contributor is a one-line install.

- A personal agent that does not care which device you're at. Voice in the kitchen, terminal at the desk, app on the phone. State lives in the substrate; the surface is just the access point.
- Long-horizon work — agents that run for days, weeks, months — that is economically impossible on hosted AI today, becoming the obvious shape for self-hosted compute.
- An ecosystem of agents, skills, and fine-tunes published to a shared substrate the way packages publish to NPM today.

We do not claim this is delivered. We claim each layer is scoped well enough that the composition is reachable from where we are now.

The falsifiable next milestone

The integration is real the day a Prithvi instance is killed on one machine, brought back on a different machine, and resumes mid-conversation — the hardware changes, the agent does not notice. That is the first milestone we will recognise. Everything before it is preparation.

Engage

- **Read** — the technical report for experiments and numbers; the architecture for the v0.1 contract; the north star for the long vision.
- **Run** — the README walkthrough takes under an hour from "Python + a GGUF" to "the right token comes back."
- **Push back** — issues at https://github.com/fthrvi/nakshatra/issues, email at tankaifish@gmail.com, or community-staked conviction at pnl.market.

The dream is big. The architecture is small. The work is the work.