LatentWorlds AI

THE DEPLOYMENT BACKEND FOR PHYSICAL AI

Hiring now
2026-02-24

Senior Software Engineer

Distributed Systems

Type : Full-time | Location : Delft, Zurich, or remote

Apply : Send your CV + short note + relevant work links to careers@latentworlds.ai

LatentWorlds builds the **deployment backend for physical AI**. We turn fleet hours into compounding autonomy by making robotics data reliable to collect, usable to retrieve, and practical to turn into training and debugging artifacts.

This role is focused on **DataCore**: the Rust backend that ingests multi-modal recordings from robots, indexes them with robotics semantics, and serves synchronized slices fast enough for daily debugging and iteration.

The Role

You will build core distributed systems that sit on the **critical path of robot deployments**. The hard part is not store-some-logs plumbing; it is making capture and retrieval dependable under real constraints: unstable networks, large payloads, time alignment across sensors, and customers who need clear guarantees.

We are at an early stage and the scope is **intentionally flexible**. As we learn from pilots, priorities shift. We want someone who enjoys owning ambiguous problems and turning them into clean primitives.

What You Will Work On

Projects vary, but you will likely touch a mix of:

- **Ingestion reliability:** resumable uploads, backpressure, durable acknowledgements, and end-to-end integrity checks.
- **Storage and index:** session and recording manifests, metadata models, and indexing that makes time-aligned slice retrieval cheap and fast.
- **Retrieval:** APIs that return synchronized slices across sensors, state, and logs with predictable latency.
- **Dataset and pipeline primitives:** reproducible exports, dataset versions, and provenance hooks that fit into existing training stacks.
- **Ops and governance:** identity, RBAC, audit trails, and retention and deletion controls tied to recordings.
- **Observability and cost:** the metrics that keep us honest (ingest success, query latency, cost-to-serve) and the tooling to regress them.
- **Front end integration** when role-critical features require user-facing workflow changes.

Why This Exists

Most robotics teams can record data. The failure mode is that nobody trusts it. Files go missing, timestamps drift, the right 20 seconds are hard to find, and datasets become a pile of scripts nobody can reproduce.

DataCore is meant to be the missing backend layer: reliable capture, coherent aggregation (sessions and manifests), and fast retrieval that makes the incident-to-fix loop practical.

What Success Looks Like

In the first weeks, success is shipping an end-to-end improvement that customers can feel, with measurements to prove it. That might be a reliability upgrade, a materially faster retrieval path, or a new primitive that makes a pilot deployment simpler.

Over the next months, success is hardening those primitives into something teams can run in production:

- Predictable ingest under bad networks.
- Slice retrieval that stays fast as data grows.
- Explicit retention and access controls.
- A clear path from recordings to reproducible dataset exports.

What We Are Looking For

We care most about **talent, ownership, and learning speed**. Frameworks change; good instincts compound. Strong candidates usually have evidence of:

- **Production experience:** shipping a production distributed system that other engineers relied on (storage, streaming, databases, infra, and similar systems).
- **Systems thinking:** comfort reasoning about partial outages, retries, backpressure, consistency, and correctness at the edges.
- **API craft:** pragmatic API and data model design that stays stable under real product pressure.
- **Clear communication:** you write down invariants, make tradeoffs explicit, and do not hand-wave reliability.
- **Fundamentals:** strong programming fundamentals. We use Rust heavily, but we value craft more than a specific language.

Bonus Points (Not Required)

- Rust experience in async and systems code.
- Multi-tenant SaaS infrastructure (auth, isolation, metering, audit).
- Protocol work (gRPC, QUIC, custom transport) and performance profiling.
- Exposure to robotics data formats (ROS/ROS2, bag files, custom log systems) and time sync.

How to Apply

Email **careers@latentworlds.ai** with:

1. Your CV or LinkedIn.
2. A short note on what you have built that is most relevant (and why).
3. Links to code, demos, writing, or a portfolio (if available).

If you are excited about the problem but do not match every bullet, you should still reach out.

Equal Opportunity

LatentWorlds AI is an equal opportunity employer. We welcome applicants of any background and identity, and we care most about the quality of your work and your ability to collaborate.