chronosphere

+

arize

Full-Stack AI Observability with Chronosphere + Phoenix

From Infra to Model Performance with OpenTelemetry

Overview

- About Arize.ai
- Why I picked Arize.ai
- Why a Partnership makes sense
- Integration Overview (Miro board)
- Beyond MVP
- Demo

About Arize.ai

- AI observability & evaluation platform – monitor and improve model quality
- Creators of Phoenix (OSS) – LLM/AI observability & eval platform
    - Prompt/response traces
    - Eval scores (toxicity, correctness, etc.)
    - Model comparisons & RAG workflows
- Maintainers of OpenInference (OSS) – OTel-aligned standard for AI tracing
    - Model comparisons & RAG workflows
- Standards-first – built on OpenTelemetry, integrates with existing observability stacks

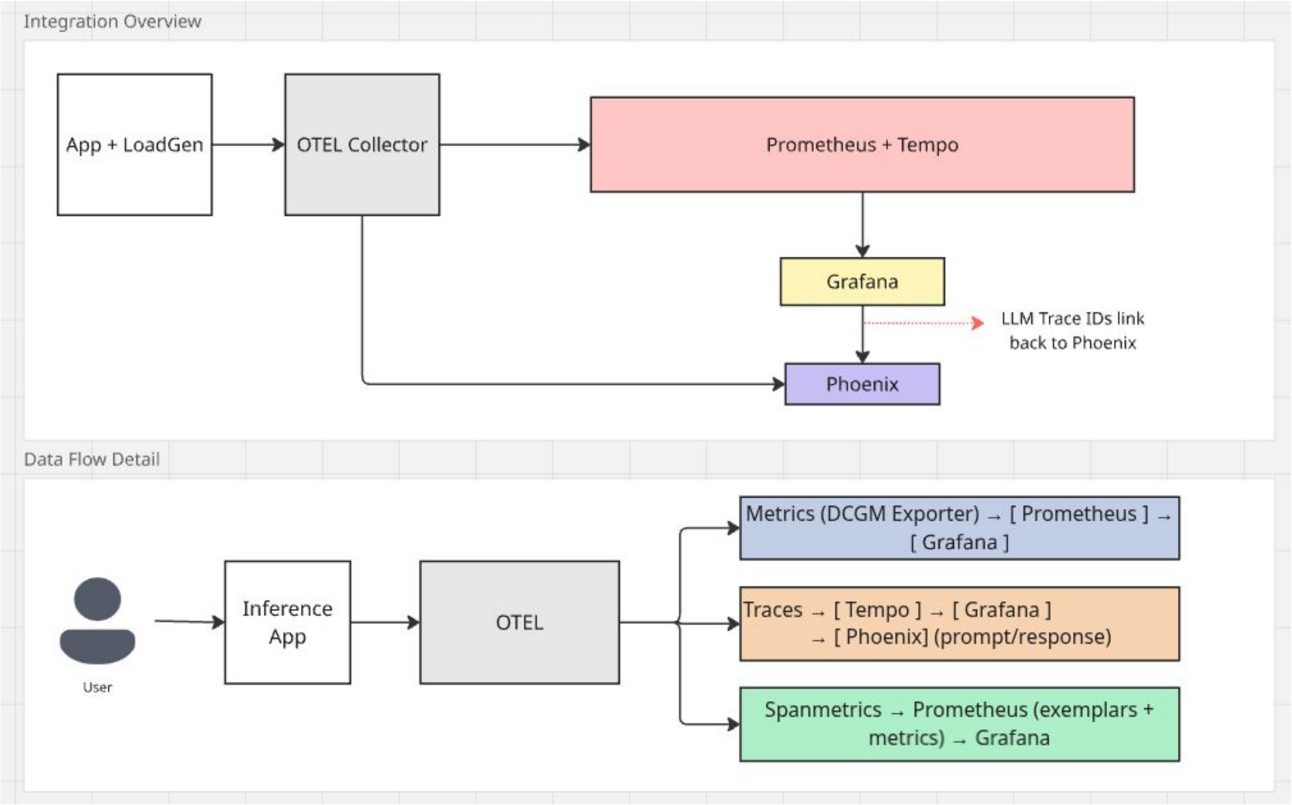
Why I picked Arize.ai

- AI adoption is exploding → Customers want more than raw infrastructure metrics.
- Teams need to answer:
    - Are my LLM outputs good — accurate, safe, and unbiased — ***not just fast and available***?
    - How do different models or versions compare in quality and reliability?
    - Are my embeddings drifting in ways that hurt answer quality?
    - How do infrastructure bottlenecks (GPU, latency, concurrency) impact inference quality and user experience?
    - When a poor response happens, can I trace it back to the root cause — data, model, or system?
- Business value: reduced cost, higher trust in AI outcomes, faster MTTR.

Why a Partnership Makes Sense

- **Chronosphere gains** → Model-aware telemetry to complement infra observability; closes any *perceived* AI gap vs. Datadog/New Relic
- **Arize/Phoenix gains** → Enterprise-grade infra correlation and scale (billions of datapoints/sec) that customers already trust
- **Customers gain** → Unified view of AI health — infra, GPUs, latency, *and* model quality in one place
- **Shared DNA** → Both built on **open standards / open source** (OTel + OpenInference + Prometheus); ethos of interoperability
- **Competitive edge** → Joint story against common competitors (DD/NR) who market "AI observability" but lack true LLM-aware depth

Integration Overview (Miro Board)



Integration Overview

App + LoadGen

OTEL Collector

Prometheus + Tempo

Grafana

LLM Trace IDs link back to Phoenix

Phoenix

Data Flow Detail

User

Inference App

OTEL

Metrics (DCGM Exporter) → [Prometheus] → [Grafana]

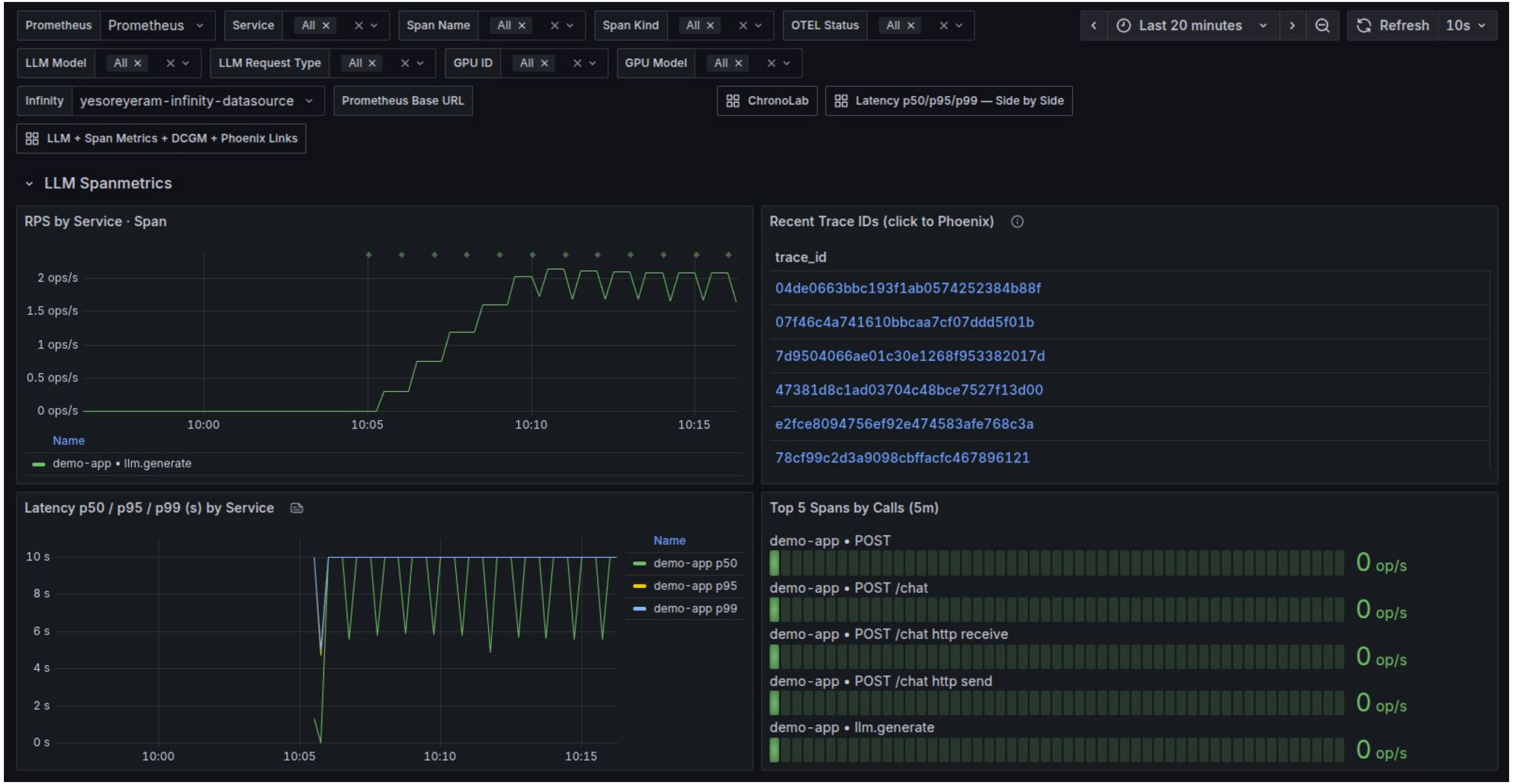
Traces → [Tempo] → [Grafana]
→ [Phoenix] (prompt/response)

Spanmetrics → Prometheus (exemplars + metrics) → Grafana

Beyond MVP

- Add Phoenix evals (toxicity, correctness, helpfulness) on captured traces
- Enable model comparisons in Phoenix to test different LLMs/prompts
- Extend tracing to RAG workflows (retrieval spans + embedding drift)
- Support multi-turn conversation traces instead of only single calls
- Integrate OpenInference SDK for standardized LLM observability
- Utilize “arize-phoenix-otel” package for easier instrumentation

Demo Time



Prometheus: Prometheus
Service: All
Span Name: All
Span Kind: All
OTEL Status: All
Last 20 minutes
Refresh 10s

LLM Model: All
LLM Request Type: All
GPU ID: All
GPU Model: All

Infinity: yesoreyeram-infinity-datasource
Prometheus Base URL
ChronoLab
Latency p50/p95/p99 — Side by Side
LLM + Span Metrics + DCGM + Phoenix Links

LLM Spanmetrics

RPS by Service · Span

Recent Trace IDs (click to Phoenix)

trace_id
04de0663bbc193f1ab0574252384b88f
07f46c4a741610bbcaa7cf07ddd5f01b
7d9504066ae01c30e1268f953382017d
47381d8c1ad03704c48bce7527f13d00
e2fce8094756ef92e474583afe768c3a
78cf99c2d3a9098cbffacfc467896121

Latency p50 / p95 / p99 (s) by Service

Top 5 Spans by Calls (5m)

demo-app • POST	0 op/s
demo-app • POST /chat	0 op/s
demo-app • POST /chat http receive	0 op/s
demo-app • POST /chat http send	0 op/s
demo-app • llm.generate	0 op/s