

ENTERPRISE PROGRAM BRIEF

LLMOps and GenAIOps Engineering

Operationalise a supplied AI application using reproducible packaging, identity and configuration management, CI checks, telemetry, bounded failure handling, and rollback. Participants work on application-level operations and identify the additional work needed for broader AI platforms.

VNode ITes

VAI-204

Coming soon

DURATION
32 hours

LEVEL
Intermediate to Advanced

FORMAT
Private cohort

AUDIENCE PROFILE

Built for these roles

DevOps engineers, cloud engineers, SREs, AI engineers, and application owners with deployment experience.

OVERVIEW

Executive overview

Operationalise a supplied AI application using reproducible packaging, identity and configuration management, CI checks, telemetry, bounded failure handling, and rollback. Participants work on application-level operations and identify the additional work needed for broader AI platforms.

READINESS

Prerequisites

- Git, CI/CD, containers or equivalent application packaging, cloud identity fundamentals, Python application operations, and a working understanding of LLM requests. A tested source application is supplied.

PROGRAM OUTCOMES

Capabilities your teams will gain

- Version application prompts, retrieval assets, configuration, and evaluation data.
- Package and release an AI application through defined gates.
- Trace failures without unnecessarily logging sensitive data.
- Measure latency, usage, and cost by workload.
- Rehearse incident response, rollback, and model-change evaluation.

ENTERPRISE CUSTOMIZATION

Enterprise customization

Tailor this program to your organization's priorities: Deploy and operate versioned AI applications with quality gates and observable costs

- 4 learning days of 8 hours, or 8 sessions of 4 hours. Learning hours exclude breaks and include module exercises and capstone time. Setup is completed before the course.

CURRICULUM

Curriculum roadmap

1 MODULE 1 · 3 PROPOSED HOURS

GenAI operating model and service objectives

An AI application changes through code, prompts, data, model versions, and platform configuration. Define who owns each asset and what useful service behaviour means. Quality objectives and incident expectations should fit the workflow rather than copy a generic infrastructure target.

- Asset lifecycle and ownership
- Service indicators and task-level measures
- Dependency and failure inventory

Planned lab: Create an operating model and service objectives for a supplied support application.

Expected output: Ownership map and service objectives.

2 MODULE 2 · 4 PROPOSED HOURS

Packaging and environment configuration

Reproducible deployment starts with known dependencies and explicit configuration. Keep secrets and environment-specific identities outside the application package. A repeatable startup and health check helps separate packaging problems from model-quality problems.

- Dependency locking and application packaging
- Environment settings and secrets
- Health checks and deployment validation

Planned lab: Package the supplied service, run it locally, and validate its configuration and health behaviour.

Expected output: Versioned application package and setup record.

3 MODULE 3 · 4 PROPOSED HOURS

Identity and controlled access

The application identity needs only the permissions required for its task. Separate management permissions from runtime access and source permissions. Review how local development authentication differs from the deployed workload identity.

- Runtime identity and least privilege
- Model and source access
- Secret rotation and configuration ownership

Planned lab: Configure the selected sandbox identity path and demonstrate an allowed call and a denied operation.

Expected output: Access inventory and verified permission boundary.

4 MODULE 4 · 5 PROPOSED HOURS

CI and release quality gates

A release should identify all changed AI application assets and run the corresponding checks. Reject a candidate on a blocking control failure even when it builds successfully. Keep model-generated scoring distinct from deterministic tests in the report.

- Build and behaviour checks
- Evaluation integration and approvals
- Artifact promotion and release records

Planned lab: Build a pipeline that packages a candidate, runs tests and evaluations, and records an explicit release decision.

Expected output: Pipeline and release evidence.

5 MODULE 5 · 4 PROPOSED HOURS

Observability and operational diagnosis

Useful telemetry connects a user task with application steps, model calls, and dependencies. Record the minimum information needed to diagnose failures. Correlation and version fields help identify whether a regression followed a prompt, source, or deployment change.

- Traces, correlation, and version metadata
- Latency and token usage
- Telemetry minimisation and retention choices

Planned lab: Diagnose a seeded failure using traces and build a small operational dashboard.

Expected output: Trace walkthrough and dashboard specification.

6 MODULE 6 · 4 PROPOSED HOURS

Reliability and recovery

Bound retries and distinguish safe reads from state-changing operations. A fallback must preserve the workflow contract and should not quietly change permissions or quality expectations. Test recovery procedures with realistic failure timing.

- Timeouts, retry budgets, and idempotency
- Fallback decisions and dependency outages
- Rollback and recovery rehearsal

Planned lab: Inject model unavailability and a failed write, recover safely, and revert a candidate version.

Expected output: Failure test record and runbook.

7 MODULE 7 · 4 PROPOSED HOURS

Cost performance and model change

Cost per request is only part of the operating picture. Account for retrieval, hosting, retries, and review effort where relevant. Compare model or context changes against quality and latency evidence before adopting a cheaper or faster configuration.

- Cost allocation and workload budgets
- Context and caching trade-offs
- Model upgrades and regression checks

Planned lab: Compare two permitted configurations using a fixed workload and document quality, usage, latency, and excluded costs.

Expected output: Cost-performance comparison and upgrade plan.

8 MODULE 8 · 4 PROPOSED HOURS

Operations capstone

Release and operate a bounded application candidate using the evidence assembled in earlier modules. Demonstrate a failure and recovery, then hand the service to another operator using only the runbook.

- Candidate release and incident rehearsal
- Evidence and ownership review
- Operational handover

Planned lab: Deploy the supplied application to the selected sandbox path, run the incident drill, and present a handover.

Expected output: Release package and operating handover.

CAPSTONE

Capstone

A versioned support AI service with release gates, traces, cost assumptions, and a demonstrated rollback and recovery drill.

ASSESSMENT

Assessment

Release reproducibility 25%; identity/configuration 20%; observability 20%; failure recovery 25%; handover 10%. A missing or unsafe recovery procedure requires remediation.

DELIVERABLES

Participant deliverables

- Application deployment package
- CI and evaluation gates
- Telemetry and cost report
- Runbook and recovery evidence

SCOPE

Scope boundaries

Full GPU cluster administration, model pretraining, multi-cloud platform certification, and a guarantee of production availability. **Delivery options:** 4 learning days of 8 hours, or 8 sessions of 4 hours. Learning hours exclude breaks and include module exercises and capstone time. Setup is completed before the course.

DELIVERY

Delivery options

4 learning days of 8 hours, or 8 sessions of 4 hours. Learning hours exclude breaks and include module exercises and capstone time. Setup is completed before the course. Proposed hours exclude breaks and are subject to pilot validation.

PRIVATE DELIVERY

Plan a private cohort for your team

Scope delivery format, role mix, and rollout timeline with our solutions team.

[Plan Private Delivery](#)[Book advisory call](#)