Problem Statement – Smart Factory AI LLMOps Platform

Title: Build a Full-Stack LLMOps Platform for Smart Factory AI

Context / Background

Manufacturing organizations adopting LLM assistants need an end-to-end platform that goes beyond model training. They must monitor usage and cost, assess response quality and drift, manage prompts/datasets, route human review, alert on anomalies, audit every action, and configure retrieval/generation settings—all while maintaining privacy, security, and scalability. Your mission is to design and implement a complete LLMOps platform replicating the Smart Factory AI dashboard and its pages, covering observability, evaluation, governance, and configuration.

Summary (What You'll Build)

Create a modular, production-ready LLMOps solution that ingests logs from an LLM-powered factory assistant, computes key KPIs, detects quality drift, and surfaces them in interactive dashboards. The system should include trace/session explorers, evaluator management, prompt versioning, dataset evaluation, annotation queues, alerting, audit logs, and advanced settings (retrieval, embedding, generation, hybrid search, guardrails, project). It must support AI-generated root-cause analysis (RCA), provide human-friendly recommendations, and be extensible for new models and datasets.

Objectives / Goals (Success Criteria)

Objective	Description	Evidence / KPIs
A. Observability & Cost Dashboard	Build a real-time dashboard showing total traces, total cost, total tokens, and average latency. Display user satisfaction, task-completion rate, first-response accuracy, escalation rate, daily active users, response-quality distribution, and user feedback counts. Include charts for traces by name, cost by model, evaluation scores,	Dashboard loads within 2 s; KPIs update hourly; accuracy verified against source logs.

	model-usage details, and issues reported.	
B. Trace & Session Management	Develop pages to list all traces (timestamp, name, input, latency, tokens, cost, hallucination/context/concision scores) and sessions (session-ID, user, trace count, total tokens, total cost, created date). Provide search/filter, pagination, and drill-down.	100% of traces and sessions displayed; search returns results < 1 s; UI responsive.
C. Evaluators & Templates	Implement an evaluator registry showing active evaluators (status, template, score name, target, sampling). Allow creation/editing of evaluation templates with guidelines and variables (e.g., hallucination detector, context relevance, conciseness). Record evaluation logs (timestamp, evaluator, trace ID, score, duration, status).	Able to register/disable evaluators; logs persist in Delta; evaluation latency < 1 min.
D. Prompt Library & Versioning	Create a prompt management module listing prompts (e.g., Factory Assistant System, Safety Issue Classifier). Support editing prompt text, guidelines, detected variables, versioning, environment (prod/test), and change history.	Version history retained; switching versions updates prompt used in evaluation; UI indicates prod vs. test.

E. Dataset Management & Evaluation	Develop a datasets module listing evaluation datasets (e.g., Q&A Golden Set, Safety-Critical Scenarios, Multi-Hop Reasoning). Provide item view (input, expected output, verification status) and run evaluation tests to compute pass rate, average score, average latency, total cost, and breakdown by metrics (hallucination, context relevance, conciseness, correctness). Plot performance and latency trends over time.	Evaluation results stored; pass/fail thresholds configurable; trend charts update automatically.
F. Annotation Queues (Human-in-the-Loop)	Implement an annotation queue system for manual review. Example queues: High-Risk Responses (low confidence/safety flags) and Training Dataset Curation. Track completed vs. pending items, progress, score configurations; allow queue processing and creation of new queues with custom scoring.	Queue UI supports start/stop tasks; progress updates; scores recorded; 100% coverage of flagged items.
G. Alerting & Rules	Provide an alerts dashboard summarizing active alerts (total, critical, warnings) and listing each alert (type, severity, metric, threshold, current value, time). Allow creation of alert rules (e.g., context relevance threshold, daily cost limit, P95 latency threshold). Include actions to acknowledge/mute and show rule history.	≥3 alert rules; alerts fire within 5 min of breach; actions recorded in audit logs.

H. Audit Trail	Build an audit log capturing timestamp, type (evaluator, annotation, prompt, dataset), action (e.g., Evaluator Run Completed, Annotation Submitted), user, and details. Support filtering, search, and CSV export.	100% of relevant actions logged; logs immutable; export succeeds.
I. Settings & Guardrails	Develop a settings interface with tabs for Retrieval (top-k, similarity threshold, chunk size/overlap, semantic ranker, reranker model), Embedding (model, dimensions, batch size), Generation (model, temperature, max tokens, top-p, penalties), Hybrid Search (BM25 and vector weights), Guardrails (minimum confidence, hallucination sensitivity, toxicity filter, PII detection, max context length, auto-escalation threshold), and Project (name, environment, API key, webhook URL, telemetry simulator with spike frequency and noise level).	Setting changes persist; toggles/thresholds apply to downstream evaluations; telemetry simulator adjustable.
J. Drift & Hallucination Monitoring	Compute drift metrics (e.g., context relevance vs. baseline, hallucination rate change, retrieval overlap). Trigger drift-detection alerts and produce AI-generated RCA with recommendations (e.g., retrain model/update embeddings). Show drift metrics and recommendations in the dashboard.	Drift detection accuracy $\geq 85\%$; RCA ≤ 2 sentences; recommendations actionable.

K. User Feedback & Evaluation Analyzer	Ingest user feedback (thumbs-up/down, free-text), categorize into positive/negative, extract common issues (e.g., incomplete specs, outdated procedures, missing safety warnings). Compute evaluation scores (hallucination, context relevance, conciseness, correctness) and display aggregated results.	Sentiment classifier ≥ 80% accuracy; issue extraction identifies top 3 issues; evaluation scores reproducible.

Scope Clarifications

In-scope: frontend dashboards; backend APIs and data models; evaluation and drift detection; prompt/dataset/annotation management; alerting; audit logs; settings; optional LLM API for RCA summaries (strict non-retention); deployment via GitHub Actions + Terraform.
Out-of-scope: training large language models from scratch; <1 s real-time streaming; cross-tenant identity/billing.

Tech / Tooling Expectations

- Data & Compute: Databricks or Snowflake; Delta Lake; ADF/Databricks Workflows.
- Monitoring & Logging: Python (pandas/PySpark); Prometheus/Grafana or Azure Monitor; MLflow for prompt/evaluator versioning; Great Expectations for data quality.
- AI/ML: OpenAI/Azure OpenAI/Anthropic for RCA summaries and evaluation; scikit-learn/statsmodels for drift; spaCy/regex for sentiment/issues.
- Frontend: Streamlit/React or Power BI; Tailwind for styling.
- CI/CD & IaC: GitHub Actions; Terraform; OIDC for secretless auth.
- Security & Governance: OAuth/OIDC; Key Vault/env vars; PII redaction; guardrails.

Deliverables

1. Modular platform codebase with README and architecture diagram.
2. Delta schemas for traces, sessions, evaluations, prompts, datasets, annotations, alerts, audit logs, settings.
3. Dashboards & UIs replicating all Smart Factory pages and KPIs.
4. Evaluation and drift modules with MLflow-tracked experiments and reproducible metrics.
5. Alerting & notification system with configurable rules.
6. Annotation queue interface with scoring.
7. Audit log module with export function.

8. Configuration interfaces for retrieval, embedding, generation, hybrid search, guardrails, project.
9. Deployment scripts (Terraform + CI/CD) and a 5–7 min demo video.
10. Documentation (setup guide, data dictionary, API contracts, evaluation rubric, limitations).

Evaluation Criteria

Criterion	Weight	What We Look For
Feature Completeness & Correctness	25%	All modules implemented and match KPIs; data flows correct.
Observability & UI/UX	20%	Dashboards are intuitive, responsive, and update near real-time.
Automation & DevOps	20%	Terraform IaC; CI/CD passes; tests/linters; zero secrets in code.
AI & Evaluation Quality	15%	Accurate drift/evaluation metrics; clear AI-generated RCA.
Scalability & Resilience	10%	Handles growth; idempotent merges; retries.
Documentation & Demo	10%	Clear README, runbook, diagrams, polished demo.

Additional Guidelines & Tips

- Security & Privacy: All keys in Key Vault; no PII in logs; least-privilege RBAC.
- AI Safety: Exclude sensitive data from prompts; log prompts/responses in MLflow; provide offline fallback.
- Modularity: Each module (dashboard, evaluation, datasets, prompts, annotation, alerts, audit, settings) is reusable and documented.
- Test Data: Use synthetic/anonymized logs; seed datasets; create drift test cases.
- Extensibility: Use config files (datasets.yml, alert_rules.yml, settings.yml) to onboard without code changes.
- Collaboration: Branching strategy (feature → dev → main); PR reviews; coding standards.