LLMOPS.SI · PRODUCTION READINESS

LLMOps Production Readiness Checklist

50 things to verify before you put an LLM in front of real users.

50 checks across the 8 layers of the LLMOps stack · v2026.1 · llmops.si/checklist

/ 01 Prompt management

- [ ] We can roll back a prompt, model or config in one step
- [ ] Every prompt and config change is versioned
- [ ] Prompt changes are tested against an eval set before production
- [ ] System prompts are stored outside application code
- [ ] We can attribute a quality change to a specific prompt version
- [ ] Prompt templates are reviewed like code (PR + approval)

/ 02 Evaluation

- [ ] We have an eval dataset that reflects real traffic
- [ ] We measure accuracy or task success, not just vibes
- [ ] Evals run automatically on every prompt or model change
- [ ] We track hallucination / faithfulness rate
- [ ] We test safety and refusal behaviour
- [ ] Releases are gated on eval results
- [ ] Eval failures page or block the deploy

/ 03 Observability

- [ ] We log every request and response
- [ ] We capture full traces for multi-step chains and agents
- [ ] We measure p50 / p95 latency in production
- [ ] We track token usage per request and per feature
- [ ] We alert on error-rate and latency regressions
- [ ] We can search and inspect individual production traces
- [ ] We can replay a production request to reproduce a failure

/ 04 Cost control

- [ ] We have budget alerts on token spend
- [ ] We know our cost per request and per active user
- [ ] Prompt / context caching is enabled where it helps
- [ ] We cap max tokens and context size per request
- [ ] Easy traffic is routed to smaller, cheaper models
- [ ] Spend is attributable to a team, feature or customer

/ 05 RAG operations

- [ ] We measure retrieval quality, not just final answers
- [ ] We have a fallback when retrieval returns nothing relevant
- [ ] Chunking and embedding choices are evaluated, not assumed
- [ ] The index is refreshed on a known schedule
- [ ] We detect and handle stale or missing context
- [ ] We monitor embedding / data drift over time

/ 06 Security

- [ ] We treat all user and retrieved input as untrusted
- [ ] We have guardrails for PII and data leakage
- [ ] Secrets and keys are never exposed to the model context
- [ ] We test for prompt injection and jailbreaks
- [ ] Tool and data access is scoped with least privilege
- [ ] Outputs are validated before they trigger actions

/ 07 Governance

- [ ] A human reviews critical or high-risk use cases
- [ ] We have an incident process for model failures
- [ ] High-risk actions require an explicit approval step
- [ ] We keep an audit trail of inputs, outputs and decisions
- [ ] We can explain and reproduce a given production decision
- [ ] Data retention and privacy policy is defined and enforced

/ 08 Deployment

- [ ] We can revert a release without a code redeploy
- [ ] LLM changes go through CI before production
- [ ] We have a staging environment that mirrors production
- [ ] On-call knows how to triage an LLM incident
- [ ] Changes roll out progressively (canary / percentage)
- [ ] Model and provider fallbacks are configured