LLM Workflow Build & Harden

Phil Stevens
philipstevens.github.io

Implement eval gates and a regression harness, then harden the workflow until it meets the production bar.

SCOPE	One LLM workflow with an agreed production bar, hardened to meet quality, safety, and operational targets.
PRIMARY OUTPUT	Eval harness in CI + hardened workflow + release gates + rollback plan (ready to ship).

Best Fit

- You want this workflow shipped with predictable behavior on real inputs.
- You need eval gating in CI to prevent regressions and unsafe releases.
- You have enough examples and access to iterate quickly.

Inputs Required

INPUT	MINIMUM	WHY IT MATTERS
Agreed production bar	Workflow spec with acceptance criteria (from Audit or equivalent)	Defines what "done" means so hardening has a clear target.
Example set	50+ representative examples; 10–30 must-pass candidates	Prevents "happy-path only" evals; makes gates meaningful.
Integration access	Repo access or endpoint to invoke the workflow in staging	Allows harness integration, CI gating, and reproducible runs.
SME review bandwidth	Someone who can review outputs and approve changes weekly	Enables fast iteration and prevents quality drift during hardening.

How the Work Runs

PHASE	WHAT HAPPENS
0. Contracts + baseline	Enforce output contracts, encode refusal/escalation policy, define trace fields, establish baseline.
1. Eval harness + CI gates	Build case runner, integrate must-pass blocking gate, add automated checks to CI.
2. Hardening loops	Iterate on prompts, RAG tuning, tool controls, and cost/latency until gates pass consistently.
3. Release readiness	Define rollout plan, test rollback procedure in staging, finalize monitoring checklist.

Timeline varies by integration complexity and number of failure surfaces (RAG/tools/agents).

What usually moves the needle

- Prompt restructuring vs prompt expansion
- Moving logic out of the model and into validators
- Shrinking must-pass sets instead of bloating evals
- When *not* to fine-tune

Deliverables

ARTIFACT	PURPOSE
Eval harness integrated into CI	Runs tests on every PR so regressions never reach production unnoticed
Test sets (golden, edge, regression, safety, high-stakes)	Covers critical paths and known failure modes so you catch real problems, not just happy-path passes
Must-pass gate configuration	Defines which failures block merges and releases so bad changes can't slip through
Output contract enforcement (schema + parser)	Ensures every response is valid or explicitly fails
Trace implementation (versions, tool calls, retrieval IDs)	Makes outputs debuggable and auditable
Release checklist and rollback plan	Documents rollout steps and tested recovery procedure

Definition of Done

- ✓ Eval harness runs in CI with a must-pass subset that blocks merges and releases; no untested must-pass cases.
- ✓ Coverage includes all agreed critical failure modes and representative data slices, with no known gaps.
- ✓ Workflow meets all defined thresholds (quality, safety, cost/latency) with documented evidence.
- ✓ Release checklist and rollback plan exist, are tested at least once in staging, and are signed off.

Boundaries

- Net-new product feature design is out of scope (this is reliability and shipping operations).
- Large re-architecture or platform build can be scoped separately if needed.

Next Step

Start with an Audit or provide a workflow endpoint + example set to begin Phase 0. Fastest path: contract → harness/CI → hardening loops → release ops.

Book an intro call:

calendly.com/philipstevens4/intro

Scan to book a call

Email: philipstevens4@gmail.com
Web: philipstevens.github.io

Not ready? Get the free Production Readiness Checklist to self-assess first.