Build & Harden

Phil Stevens
philipstevens.github.io

Implement eval gates and regression protection, then harden the workflow until it meets reliability, safety, latency, and cost targets.

Best Fit

YOU HAVE

- A workflow that produces useful output but drifts when prompts/models/docs change.
- Known failure modes (hallucinations, wrong routing, tool errors, retrieval regressions).
- Pressure to ship without a credible "proof" of readiness.

YOU WANT

- PR/CI gates that block must-pass failures automatically.
- Traceability (versions + evidence) so outputs are debuggable and auditable.
- A release process with rollout and rollback triggers tied to real metrics.

Inputs Required

INPUT	MINIMUM	WHY IT MATTERS
Integration surface	Repo access or an endpoint to invoke the workflow in staging	Allows harness integration, CI gating, and reproducible runs.
Examples	50+ representative examples; 10–30 must-pass candidates	Prevents "happy-path only" evals; makes gates meaningful.
RAG/tool details (if applicable)	Retrieval config, chunking rules, tool schemas, tool allow-list	Enables proxy retrieval tests, citation checks, tool correctness checks.
Operational targets	Latency p95 + cost p95 targets; safety constraints	Lets you gate performance and prevent cost regressions.

Delivery Phases

PHASE	WHAT'S IMPLEMENTED	DEFINITION OF DONE
0. Contracts + baseline	• Output contract enforcement (schema/format + strict parser) • Refusal/escalation policy encoded in output • Trace fields defined (versions, retrieval/tool calls, latency/cost)	• Every response is contract-valid or explicitly fails • Baseline run artifact saved as last-known-good
1. Eval harness + CI gates	• Case format + runner (local + CI) • Must-pass blocking gate • Automated checks (schema, refusals, citation integrity, expected-source proxy)	• PRs fail when gates fail • Run artifacts stored for diffing against baseline
2. Hardening loops	• Prompt packaging + versioning + routing decomposition • RAG tuning (context budgets, relevance thresholds, refusal on weak evidence) • Tool allow-list + argument validation + deterministic fallbacks • Cost/latency controls (token budgets, caching, model routing)	• Must-pass failures = 0 across repeat runs • Risk-tier gates met for quality/safety/perf
3. Release readiness	• Rollout plan (shadow/canary/ramp) • Rollback triggers + rollback procedure tested in staging • Monitoring checklist for first 24h	• Release review doc completed with evidence • Team can ship changes repeatedly with gates

Timeline varies by integration complexity and number of failure surfaces (RAG/tools/agents). The phases above are the delivery order because they minimize wasted iteration.

What You Get (Repo Artifacts)

Eval Suite + Gating

- Case set: golden + edge + regression + safety + high-stakes subsets
- Must-pass set that blocks PRs/releases
- Gate thresholds for quality, safety, latency, and cost
- Run artifacts stored and comparable to baseline

Production Hardening

- Strict output contract enforcement (no "best effort" drift)
- Traceability: prompt/model/retrieval versions + tool calls + retrieval IDs
- RAG/tool controls: allow-lists, validation, failure handling
- Performance controls: token budgets, caching, routing strategy

Definition of Done

✓ **Reproducible:** you can identify the exact model/prompt/retrieval config that produced any output.

✓ **Gateable:** must-pass failures block merges/releases automatically.

✓ **Safe uncertainty:** the system refuses/escalates instead of guessing when evidence is weak.

✓ **Operable:** logs/traces are sufficient to debug incidents quickly without re-running blindly.

✓ **Rollbackable:** you can revert to a prior version in minutes with explicit triggers.

Example: Minimal Gate Thresholds

```
must_pass_failures: 0
schema_validity: >= 99.5% (Tier 2) / 100% (Tier 3)
grounded_only_citation_integrity: 100% on must-pass
latency_p95_ms: <= target
cost_p95_usd: <= target
```

Thresholds should be set from baseline and ratcheted upward as the workflow hardens.

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