

CONF42 FEATURE FLAGS 2026

Beyond On/Off: Using Quality Gates to Safely Release GenAI Features

Rajesh Lingam · AI Software Developer — GenAI Evaluation & Engineering ·
Adobe Inc.



The views expressed in this presentation are my own and do not necessarily reflect those of Adobe Inc.



THE CORE THESIS

**Feature flags
manage
exposure.**

**Quality gates
manage
confidence.**

GenAI changed what “healthy” means

A system can be operationally green while answer quality quietly regresses.

- HTTP 200 does not mean the answer is grounded.
- Latency can look normal while safety or factuality drops.
- Prompt and retrieval drift can change behavior with no code change.

Flags answer “Can we expose this?”
Gates answer “Should we expose this?”



WHY ON/OFF IS NOT ENOUGH

Flags control exposure — not the quality being exposed

Feature flags are necessary. They are not sufficient.

Feature flags give you real controls

- Gradual rollout and segment targeting
- Kill switches and fast rollback
- A/B experimentation

Great for exposure control.

The gap, in one example

A prompt change improves tone for common requests, but creates grounding failures on a small set of edge cases. Operational monitoring looks fine.

A flag can roll it back after detection. A gate can catch it before exposure increases.



THE RELEASE CONTRACT

A quality gate turns opinion into evidence

The release meeting becomes a repeatable decision instead of a confidence conversation.

IF

**quality evidence meets
the release bar**

→ **increase exposure**



ELSE

**block, hold, rollback,
or require human review**

The decision is explicit, auditable, and
reproducible.



GATE DIMENSIONS

Do not collapse GenAI risk into one score

An average can hide the exact failure that matters.

Answer quality

Useful + correct?

Grounding

Supported by source?

Safety

Zero critical failures?

Regression

Worse than baseline?

Latency

Within UX budget?

Cost

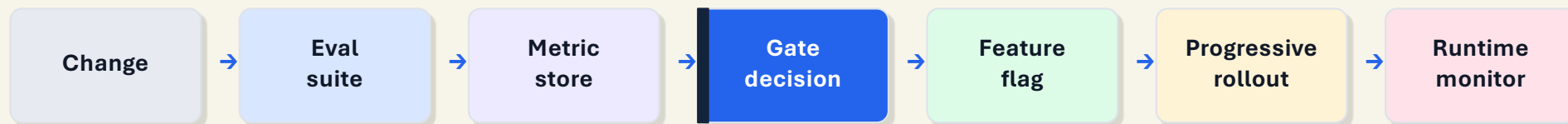
Sustainable at scale?

A useful gate tells you which dimension failed — and what to fix next.

REFERENCE ARCHITECTURE

Evaluation sits inside the PR-to-rollout loop

Every change should be treated as a releasable artifact: code, prompt, model, retrieval, or tool.



Feedback loop: runtime signals feed future gate decisions, thresholds, and golden scenarios.



THREE GATES FOR SAFE ROLLOUT

Change, release, and runtime — each at the right cost

Not every change deserves the same evaluation depth.

QG-1

Change-level

Before merge / early CI

Fast checks for obvious quality, safety, or latency regressions.

QG-2

Release-level

Before broad rollout

Representative scenarios, baselines, calibrated thresholds, and human sampling.

QG-3

Runtime-level

During progressive rollout

Telemetry, sampled evals, feedback, rollback or exposure increase decisions.



THE CENTER OF GRAVITY

“Enough” means enough evidence for the risk

The wrong answer is “run everything.” The better answer is risk-appropriate evidence.

Evaluation depth rises when either...

- User impact increases
- Failure severity increases
- Change scope expands
- Behavior is new or uncertain
- Cost of delay remains acceptable

Low-risk wording → Smoke eval

Prompt/config → Targeted scenarios

Model/RAG → Representative suite

Agentic workflow → End-to-end + guardrails

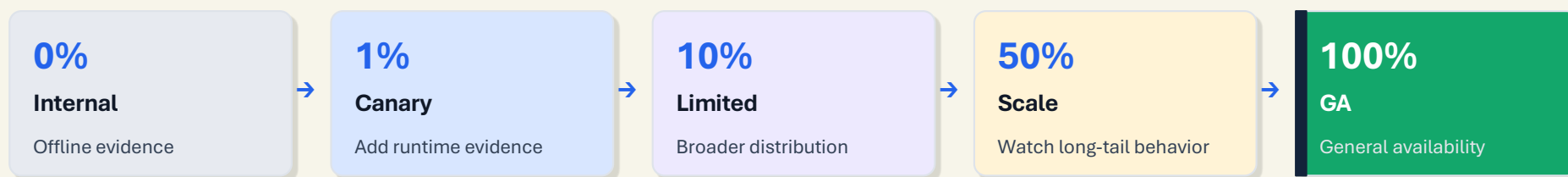
A decision framework — not a universal certification formula.



THE RELEASE LOOP

Exposure grows only as evidence grows

Feature flags become the control plane for evidence-based rollout.



- Compare to the prior stable baseline, not only absolute thresholds.
- Inspect failure categories, not just averages; pause when signals conflict.
- Make rollback a normal decision, not a failure of process.



A REGRESSION BLOCKED BEFORE ROLLOUT

The win is not “no failure” — it is failure found early

A simple story for why gates matter.

Without a gate

- Prompt improves tone; tests pass.
- System looks operationally healthy.
- Users see summaries that drift beyond source evidence.

With a gate

- Targeted eval catches retrieval-heavy failures.
- Grounding falls below the release bar → exposure stays at 0%.
- Team fixes and reruns before rollout.

The win is discovering failure before users experience it.



FOUR TAKEAWAYS

Turning a feature on is operational. Earning exposure is evidential.

The strongest release systems connect the two.

- 1** Feature flags manage exposure; quality gates manage confidence.
- 2** GenAI gates must be multi-dimensional: quality, grounding, safety, regression, latency, cost.
- 3** “Enough” is risk-appropriate evidence for the next rollout decision — not maximum test volume.
- 4** A good gate produces action: proceed, hold, rollback, or review.

Increase rollout only when the evidence earns it.



Thank You!

Rajesh Lingam · Adobe Inc.

Beyond On/Off: Using Quality Gates to Safely Release GenAI Features

Feature flags control exposure. Quality gates decide whether GenAI is safe enough to expose.