Gatling

2026

EBOOK

Improving performance decisions across the testing lifecycle with AI

How AI helps teams focus on the performance risks that actually matter

Traditional performance testing begins with a fundamental question: what should we test?

Performance failures are revenue events. Over 90% of mid-size and large enterprises estimate that a single hour of downtime costs more than $300,000, with 41% putting costs above $1 million per hour. Meanwhile, 88% of users say they are less likely to return to a website after a poor experience. This is the baseline risk that every engineering team manages, whether they realize it or not.

AI is changing how teams manage that risk, not by replacing judgment, but by improving the speed and quality of performance decisions across the testing lifecycle. For instance, during test execution itself, AI typically plays a monitoring or alerting role. Human testers still own strategy: deciding which scenarios to simulate, how much load to apply, and which performance goals matter

For business leaders, this enables a shift from “test everything” to “test what matters” aligning performance testing effort with revenue-critical workflows rather than technical completeness.

But before you even test, ask smart questions

Most performance testing failures don’t come from bad scripts.

They come from testing the wrong things.

Teams often jump straight into scripting without clearly defining risk. The result is broad coverage, low signal, and tests that pass without answering the questions that actually matter. AI-assisted planning helps reverse this pattern by forcing clarity before a single request is simulated.

AI helps surface these answers by analyzing historical test results, recent code changes, incident history, and known traffic assumptions. For example, it may flag a recently modified payment service that has never been tested at peak load, or highlight retry logic that frequently appears in error logs but is missing from test scenarios.

BEFORE WRITING A SIMULATION, TEAMS SHOULD BE ABLE TO ANSWER:

What performance failure would hurt the business most right now?

Which user flows drive the most revenue or operational load?

Where are we blind—retry paths, edge cases, or degraded dependencies?

What does “acceptable performance” mean for this release?

The risks of poorly designed tests

AI can significantly improve how teams plan, analyze, and prioritize performance tests, but only when it is used deliberately.

These trade-offs are not hypothetical. The World Quality Report 2025 found that 89% of organizations are now piloting or deploying AI-augmented workflows in quality engineering, but only 15% have achieved enterprise-scale deployment.

The table below highlights where AI adds leverage in performance testing, and where teams need to stay in control to avoid trading speed for accuracy, or insight for convenience.

AREA	AI HELPS WHEN	WATCH OUT WHEN
Speed	Test creation, analysis, and reporting are automated, shortening feedback loops	Teams stop reviewing AI output and skip validation
Accuracy	AI detects subtle regressions and risk patterns humans miss	Models are trained on incomplete or biased data
Prevention	Performance risks and SLA breaches are surfaced earlier	Predictions misinterpret unusual but valid traffic
Cost	Smarter test selection and capacity planning reduce waste	Optimization prioritizes cost over resilience
Signal vs. noise	Meaningful anomalies are surfaced, reducing alert fatigue	Poor tuning generates excessive or low-value alerts
Alignment	Clear summaries make performance visible to all teams	Oversimplification hides important technical context

AI as a guide: strategic pre-test planning

Before a single load test runs, AI can help teams decide what needs testing and why.

In the planning and design phase, teams determine which scenarios to test and create scripts or configurations.

Traditionally, this meant starting from scratch—manually coding simulations or recording user flows, which is time-consuming and prone to omission. AI speeds up this stage by automating test creation and highlighting coverage gaps:

AI CAPABILITY	OPERATIONAL IMPACT	BUSINESS IMPACT
AUTOMATED TEST CREATION	Generates a baseline load test from a prompt or specification. Engineers can describe an API or user scenario in natural language and receive a runnable test script as a starting point.	Reduces setup time and eliminates blank-page friction, allowing teams to focus on refining scenarios rather than writing boilerplate.
SMART TEST DESIGN GUIDANCE	Analyzes existing test assets and historical data to identify coverage gaps, such as high-traffic endpoints without tests or missing spike scenarios.	Improves coverage quality by surfacing blind spots early, before they become production incidents.
PROTOCOL AND SCRIPT OPTIMIZATION	Provides scripting guidance based on SDK capabilities, helps correlate dynamic values, and highlights potential errors in test scripts.	Reduces trial-and-error during script development and promotes more consistent performance testing practices.

> **The Gatling AI Assistant for IDEs enables engineers to generate simulations directly from a natural language prompt without leaving their development environment**. Describe an API or user scenario, and get a runnable test script as a starting point. But generation is only the beginning. The AI Assistant also explains and improves existing Gatling code in context, helping teams understand and refine inherited or complex simulations. For organizations migrating from legacy tools, it can convert scripts—such as LoadRunner scenarios—into modern Gatling tests, removing one of the biggest barriers to adoption.

AI as an analyst: turning results into decisions

AI's value during test execution is not about making decisions in real time; it's more about keeping test design aligned as systems evolve.

Performance results are dense, noisy, and easy to misread—especially when teams compare runs across environments, releases, or traffic profiles. AI-assisted analysis helps by automating comparison, highlighting meaningful changes, and summarizing impact in plain language.

This creates a non-expert reading layer so product managers, engineering leaders, and operations teams can understand and act on performance data without needing a dedicated metrics interpreter.

AI CAPABILITY	OPERATIONAL IMPACT	BUSINESS IMPACT
RUN-TO-RUN COMPARISONS	Generates summaries highlighting meaningful differences between test executions, such as latency increases or error spikes during specific traffic phases.	Speeds up regression detection and reduces manual analysis effort, enabling faster release decisions.
SLA-AWARE SIGNAL DETECTION	Surfaces breaches tied to defined service-level objectives while filtering out low-impact metric noise.	Keeps performance discussions focused on customer experience and business-relevant thresholds.
HISTORICAL CONTEXT AND BASELINING	Uses historical run data to determine whether current behavior deviates from established norms.	Improves confidence in performance trends and reduces the risk of overlooking gradual degradation.
SHARED UNDERSTANDING ACROSS TEAMS	Produces plain-language summaries that explain what changed and why it matters.	Enables product, leadership, and operations stakeholders to make informed decisions without deep technical analysis.

In Gatling Enterprise Edition, this surfaces through specific capabilities: **Run Trends** overlays new test results against historical baselines, instantly revealing drifts or regressions over time. **Multi-Run Comparison** highlights meaningful differences between executions, such as latency increases or error spikes during specific traffic phases. **AI Run Summary** provides a plain-language interpretation of results. **Run Stop Criteria** automatically halt a test when it no longer provides valid data —for example, when error rates spike or performance thresholds are consistently breached.

AI as a builder: designing and maintaining tests during execution

As systems evolve, performance tests must evolve with them. Every architectural change—whether a new service, a modified payload, a different scaling strategy, or a shift to event-driven communication—introduces new performance characteristics.

Without systematic updates, simulations become brittle artifacts that reflect yesterday's design instead of today's risk profile. AI-assisted tooling helps teams keep tests current by identifying mismatches between simulations and recent changes, suggesting updates and highlighting coverage gaps as architectures grow.

Instead of rewriting scripts from scratch, teams can iteratively refine them, keeping performance tests relevant without turning maintenance into a full-time job.

AI CAPABILITY	OPERATIONAL IMPACT	BUSINESS IMPACT
FASTER SIMULATION SCAFFOLDING	Generates a runnable load test from a simple description, such as simulating a checkout flow or a specific API under load.	Reduces setup time and eliminates blank-page friction, allowing teams to focus on refining scenarios rather than writing boilerplate.
SCENARIO ENRICHMENT	Suggests missing behaviors based on common failure patterns—such as retry storms, cache eviction effects, or uneven traffic distribution.	Improves realism of test scenarios, reducing the risk of untested edge cases reaching production.
COVERAGE GAP DETECTION	Compares recent tests with historical runs and known usage patterns to identify what is not being exercised.	Surfaces blind spots early, helping teams prevent performance regressions before release.
LOWER MAINTENANCE OVERHEAD	Detects mismatches caused by evolving systems and reduces manual rework when endpoints or payloads change.	Keeps test suites reliable over time, preventing performance testing from becoming a maintenance burden.

> **Modern performance testing doesn't happen once before release; it runs continuously inside CI/CD workflows**. AI-assisted validation helps teams detect when changes in APIs, payloads, or dependencies silently invalidate existing simulations. Instead of discovering broken tests weeks later, teams can identify drift at the moment code changes. This keeps performance testing aligned with delivery velocity, rather than lagging behind it.

AI as an advisor: prioritizing action after tests

Post-test analysis only matters if it leads to action. This is where many performance programs stall: teams detect regressions but struggle to agree on what to fix first, who owns it, or whether it justifies delaying a release.

AI-assisted recommendations help by adding prioritization, not by issuing directives. Rather than treating every regression equally, AI can weigh performance changes against business relevance.

A small increase in checkout latency during peak traffic may deserve more attention than a larger slowdown in an internal reporting job. This helps teams focus effort where impact is highest.

AI CAPABILITY	OPERATIONAL IMPACT	BUSINESS IMPACT
RISK-AWARE PRIORITIZATION	Combines SLA breaches, traffic patterns, and user impact to help teams rank issues based on relevance rather than raw metrics.	Ensures engineering effort focuses on performance risks that matter most to customers and the business.
TREND-BASED PREVENTION	Identifies recurring signals—such as repeated saturation in a dependency—that indicate systemic weakness over time.	Encourages proactive architectural improvements before incidents occur.
CROSS-TEAM FEEDBACK LOOPS	When performance signals flow automatically to developers, QA, operations, and product teams, testing outcomes influence design and capacity planning earlier in the lifecycle.	Reduces siloed decision-making and embeds performance awareness into everyday workflows.
ORGANIZATIONAL LEARNING	Produces plain-language summaries that explain what changed and why it matters.	Prevents repeated performance failures and supports continuous improvement.

To close the loop between insight and action, Gatling Enterprise Edition integrates natively with APM tools, placing load-testing metrics like latency, throughput, and error rates directly alongside infrastructure telemetry in the same observability dashboards SRE teams already use. **AI Run Summary** can also highlight recommended next actions based on detected regressions or SLA breaches. Teams can configure webhooks to automatically push test summaries or alerts to Slack or Microsoft Teams, or internal systems. This means test results flow directly into engineering workflows instead of sitting in a report no one opens.

Why performance incidents keep happening?

AI raises the bar for performance decisions; it doesn't replace performance expertise. It shifts where that expertise is applied.

Recent industry data validates a disciplined approach to AI. McKinsey's 2025 Global AI Survey found that only 6% of organizations qualify as "AI high performers" with meaningful financial impact from AI initiatives. Meanwhile, 42% of companies abandoned most AI initiatives in 2025, up from 17% in 2024.

The pattern is clear: AI delivers value when it is embedded into defined processes with clear ownership—exactly the model described above—not when it is deployed as a standalone capability.

In that sense, AI doesn't make performance testing easier; instead, it makes it less forgiving. When intent is unclear, automation scales the mistake. When intent is explicit, it scales insight just as efficiently.

The teams that benefit most won't be the ones with the most AI features enabled, but the ones that combine AI-assisted testing with clear ownership, explicit goals, and disciplined interpretation.

Performance failures still happen for human reasons. AI simply makes those reasons visible sooner —while there is still time to act.

Putting AI-assisted performance testing into practice

**The principles described in this paper are not about adding more features.
They are about embedding AI into a disciplined performance workflow.**

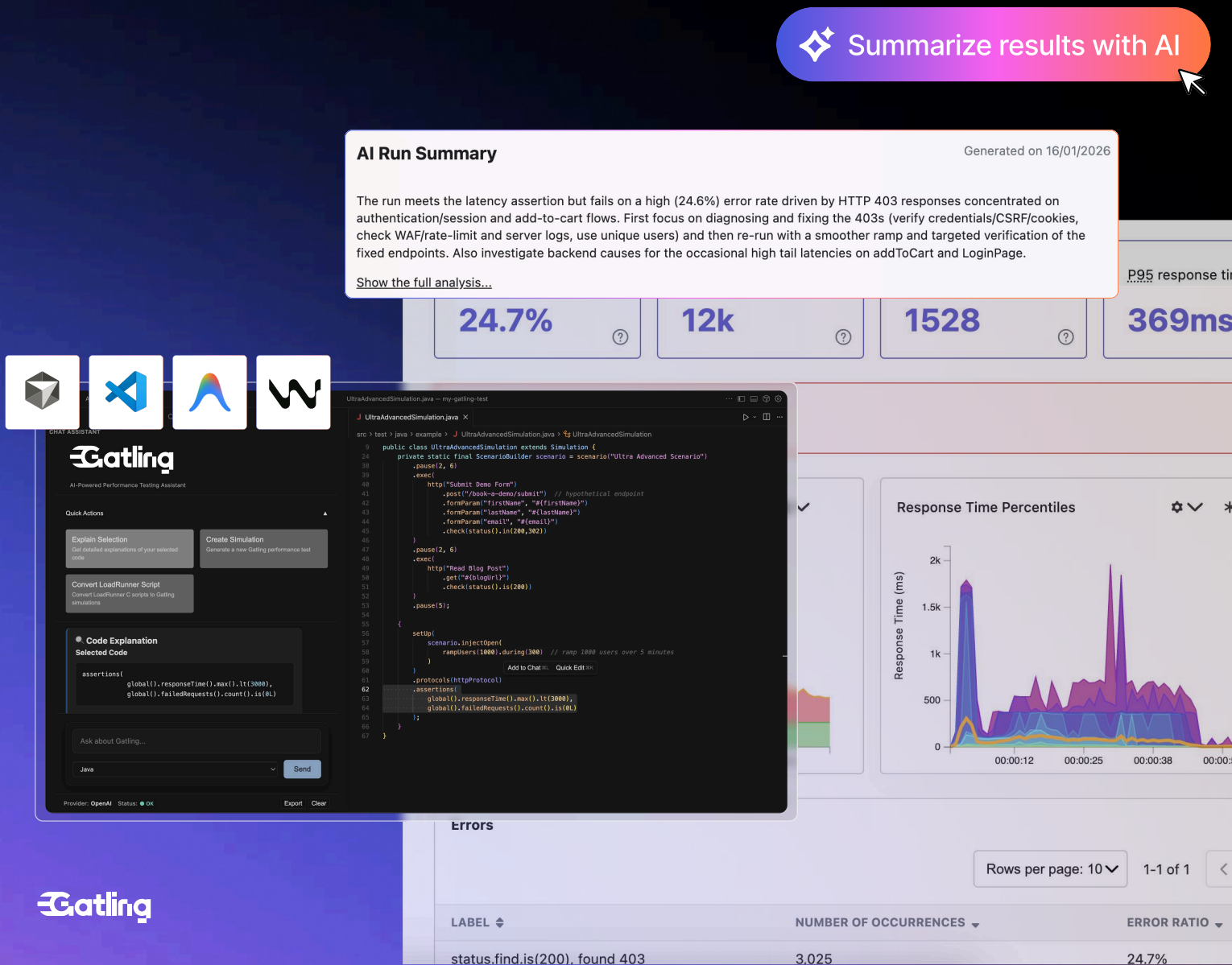
Improving decision quality across the testing lifecycle requires more than faster test creation or better dashboards. It requires continuity between planning, simulation design, result interpretation, and action.

Each stage must inform the next.

Gatling Enterprise Edition supports this lifecycle end to end. Teams can generate and refine simulations directly in their IDE using AI assistance, compare runs over time with automated summaries, define SLA-aware thresholds that surface meaningful signals, and integrate results into existing CI/CD and observability workflows.

The goal is not automation for its own sake. It is to reduce the friction between detecting a performance risk and acting on it. When AI capabilities are embedded into defined processes with clear ownership, performance testing shifts from periodic validation to continuous decision support.

That shift is what turns performance engineering from a reactive discipline into a strategic one.



Gatling

Gatling is the leading solution for modern performance engineering, enabling developers and organizations to deliver fast, reliable applications at scale.

With its powerful open-source and enterprise platforms, Gatling empowers teams to test APIs, microservices, and web apps in real-world conditions.

Trusted by thousands of companies worldwide, Gatling is the performance backbone for development, QA, and DevOps teams building the next generation of software.

Whether you're scaling APIs, migrating to the cloud, or handling flash traffic spikes, Gatling helps you deliver fast, reliable performance.

Ready to evaluate Enterprise Edition?

See how AI-assisted performance testing fits into your delivery workflows.

Talk to an expert ›