March 2026

State of AI-Powered Engineering 2026

Survey Report

Table of Contents

Introduction and Key findings

Introduction

AI adoption in software engineering has reached 90%, dramatically accelerating the speed at which software is written. AI code assistants, generate code for engineers using AI reasoning guided by natural language prompts. The usage of these tools is yielding significant benefits, with University of Chicago research[1] showing that teams that code with these tools merge 39% more pull requests than those that do not.

However, there is a clear cost to this speed. The engineering bottleneck has shifted from writing code to validating that it works. Under increasing volumes of changes, reliability has not kept pace. Google's 2025 Dora report[2] found that the adoption of AI tools correlates to an increased code instability of almost 10%. This explains why 30% of engineers report little to no trust in AI-generated code.

This context led to the emergence of the AI SRE (site reliability engineering) market. These AI tools analyze and map telemetry, codebase changes, and infrastructure signals to identify incident causes and recommend remediations.

This deficit prevents autonomous remediation from moving beyond limited, supervised use, creating a "Trust Wall" that is the impetus for this survey.

With the AIOps platform market projected to grow from $11.7 billion in 2023 to $32.4 billion by 2028 at a CAGR of 22.7%[3], this report is designed for senior engineering leaders, including VPs Directors, SREs, CTOs, and other executives who are responsible for navigating the reliability gap.

This report explores three themes:

- AI acceleration meets reliability: How enterprise engineering departments are verifying AI-generated code to ensure reliability.
- The runtime visibility gap: The importance of live execution visibility and its impact on reliability processes.
- Market perception and trust: How the current market of AI SREs is viewed and trusted to solve Production incidents, and the barriers to adoption.

Methodology

To get more insight into the state of reliability in the AI-powered engineering, we commissioned a survey of 200 SRE and DevOps leaders to shed light on their current usage of AI across the SDLC, which should solve their most pressing challenges and priorities.

This report was administered online by Global Surveyz Research, an independent global research firm. The survey is based on responses from senior-level (Directors+) DevOps Engineers, QA Engineers, Software Architects, Platform Engineers, Infrastructure Engineers, Production Support Engineers and Site Reliability Engineers (SREs) with five or more years of experience in their role. They all hail from companies with 1,500+ employees in the Finance and Software Development industries across the US, UK and EU (mix of FR/DE/IT).

The respondents were recruited through a global B2B research panel and invited via email to complete the survey, with all responses collected during Jan-Feb 2026. The average amount of time spent on the survey was 6 minutes and 43 seconds. The answers to most of the non-numerical questions were randomized to prevent order bias in the answers.

Sources:

[1] New study suggests major productivity boost when using Cursor's coding agent, via LeadDev
[2] Google Dora 2025 Report: State of AI-assisted Software Development, via DORA/Google Cloud
[3] Report: AIOps Platform Market worth $32.4 billion by 2028, via MarketsandMarkets™

Key findings

01 | Manual debugging is a major drain on engineering resources, affecting 43% of AI-generated code

Two-thirds of organizations need to manually debug 26-50% of AI-generated code changes (Figure 2). Notably, no organization can release AI code fixes with just one redeploy cycle. A total of 88% of engineering leaders require between two and three redeploy cycles, and an additional 11% require four to six cycles to verify a single change. The high volume of toil represents a significant cost, impacting finance and labor budgets. It demonstrates that despite the acceleration of software creation, AI delivery has not yet positively impacted Production reliability processes.

02 | The main bottleneck in incident resolution is runtime visibility, with 38% of developer time spent debugging, verifying, and troubleshooting

According to 60% of respondents, the biggest bottleneck in incident resolution workflows is understanding system behavior under live Production conditions (Figure 6). This visibility gap translates into developers in most companies being forced to spend an average of 38% of their time on debugging, change verification, and environment-specific troubleshooting each week (Figure 5). Since developers are one of the most valuable and expensive resources, this represents a costly waste. It highlights an urgent need to improve AI SREs and other AI coding tools.

03 | 97% of engineers rely on tribal knowledge rather than AI-assisted tools to resolve high-severity incidents

A widespread lack of confidence is a major barrier to adoption. Data shows that 77% of respondents have low levels of confidence in their current observability stack's ability to support reliable, automated code-level root cause identification and remediation (Figure 13).

On average, 54% of resolutions to high-severity incidents rely on tribal knowledge rather than diagnostic evidence obtained from AI-assisted APMs or AI SREs that automate root cause analysis and triage (Figure 7). This reliance on memory is most acute in the Finance industry, where 74% of respondents use tribal knowledge in "over 51%" of the serious incidents they face (Figure 8). As teams still do not trust AI to resolve issues, automation is frequently abandoned in serious events.

04 | 99% of engineering leaders say AI tools lack sufficient runtime visibility, with 44% of AI SRE or APM tool failures occurring due to the fact that key execution-level data was not captured.

Nearly all surveyed leaders (99%) say their AI agents operate with runtime visibility limitations. Over half (51%) report that their AI agents have limited or no visibility into live execution state, such as variables, memory, and traffic paths (Figure 12). This deficit prevents accurate root cause analysis and the generation of safe fixes. An average of 44% of AI SREs or APM tools failure was attributed to the fact that the necessary execution-level data was never captured (Figure 11).

05 | All engineering leaders say that AI SREs must be able to view live system behavior to be fully trusted

Engineering leaders agree that AI SREs cannot fix what they cannot see. According to 58% of respondents, the most important capability required to trust AI SRE remediations is the ability to provide 'evidence traces' of variables at the point of failure (Figure 16). The remaining 42% of respondents cited the ability to verify suggested fixes before they are deployed as their top 'trust-building' capability. The fact that 100% of responses coalesced around these two capabilities confirms that closing the runtime visibility gap is the key priority for SRE and DevOps leaders

Detailed report results

AI Acceleration Meets Production Reality

Validation processes have not kept pace with the speed of AI delivery. As a result, manual validation has become a primary bottleneck in software development. Because the majority of AI coding tools lack runtime visibility, engineering teams have hit a “Trust Wall”. They are forced to manually test and verify AI recommendations and resolve high-severity incidents themselves. This represents a significant barrier to scaling AI’s impact on engineering.

Engineering leaders remain hesitant about the reliability of AI code

When asked how confident they are that AI-generated code will behave correctly once deployed to Production, not a single senior engineering leader claimed to be very confident.

While 80% of the respondents express moderate confidence, 20% reported they are not very confident, while no respondents said that they were not at all confident.

These responses suggest that while leadership recognizes the value of AI-accelerated code, they remain hesitant about its predictability after Production deployment.

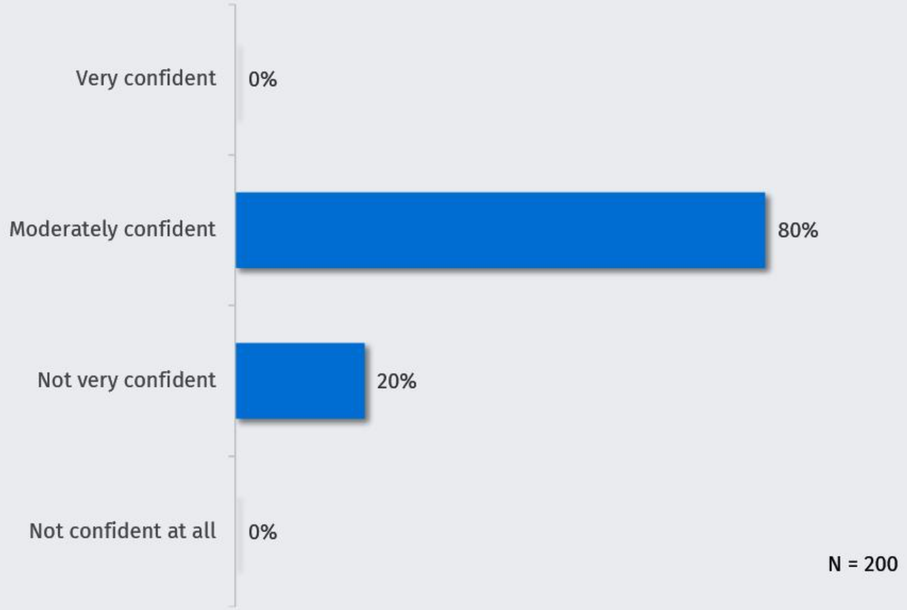


Figure 1: Confidence in Correct Behavior of AI-Generated Code Once Deployed

AI-suggested code changes require significant manual debugging in Production

AI-suggested code changes require significant manual debugging in Production

On average, 43% of AI-suggested code requires manual debugging in Production, even after passing QA and Staging (Figure 2), with two-thirds of the organizations surveyed (66%) reporting that manual debugging is still required in 26-50% of cases.

The burden is felt most keenly by those closest to Production, SREs and Production Support Engineers, 38% of whom report that this manual work is required for between 51-75% of all AI-generated changes. This is 12% higher than those holding other roles across the general survey (26% of respondents), as seen in Figure 3.

As the volume of code increases due to AI-acceleration, this is a significant bottleneck. These findings also reveal that modern Production environments are frequently not accurately reproducible in lower environments, increasing operational friction.

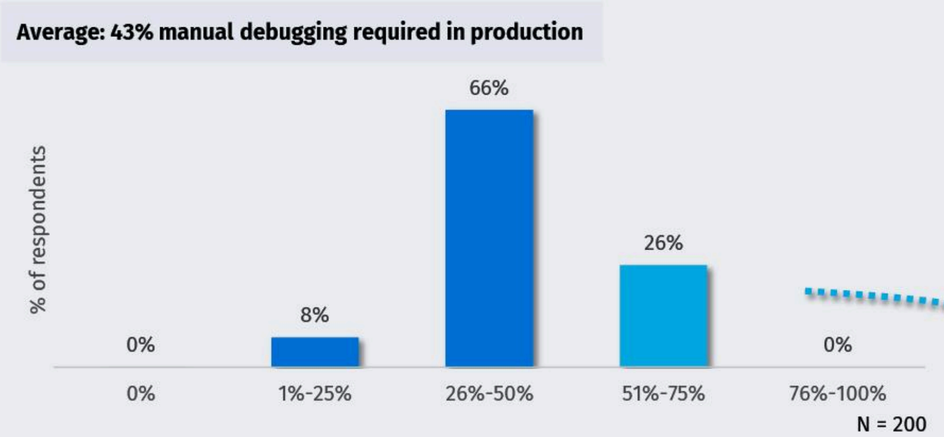


Figure 2: Percentage of AI-Suggested Code Changes That Require Manual Debugging in Production Despite Passing QA or Staging Tests

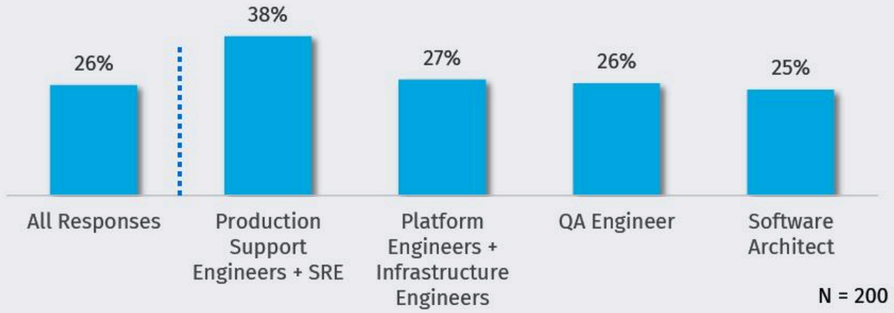


Figure 3: 51-75% Manual Debugging Required in Production by Role

It takes an average of three manual redeploy cycles to verify an AI-suggested fix in Production

The inability to verify fixes in runtime has created a high-risk "guess-and-check" workflow. On average, three manual redeploy cycles are required to verify an AI-suggested code fix in Production.

Most companies (88%) require between two to three cycles, with an additional 11% requiring four to six cycles, and 1% struggling with as many as 7-10 redeploy cycles.

Notably, none of the respondents indicated that they could release AI code fixes using just one redeploy cycle. This demonstrates that current AI SRE tools are unable to offer the autonomous resolution they promise.

This is not only a time consuming, and expensive issue, in regulated industries or large companies with various protocols and timelines, new deploys are often restricted to certain windows, with code freezes in place, making this inability dangerous.

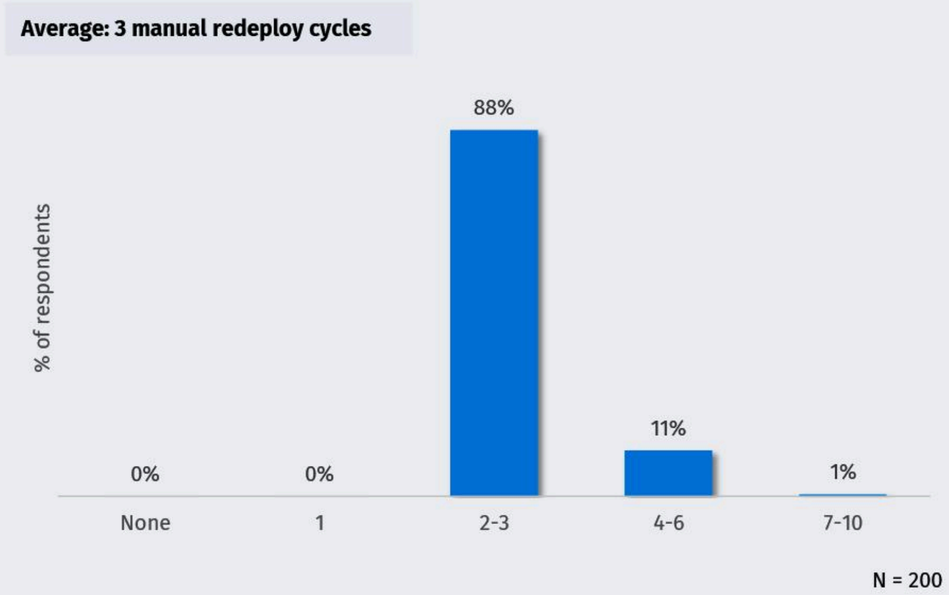


Figure 4: Average Number of Manual Redeploy Cycles Required to Verify AI-Suggested Code Fixes in Production

Manual toil consumes 38% of weekly engineering capacity

The manual validation process drains significant engineering resources. Developers now spend an average of 38% of their time each week on debugging, verification, and environment-specific troubleshooting each week.

Developers constitute one of the most valuable and expensive resources of companies, so any time they spend on tasks that detract from actual development represents a costly waste for the business. The fact that for 88% of companies this ‘reliability tax’ consumes between 26% and 50% of their developers’ week indicates that it is a major pain point which current AI innovations have not yet been able to solve.

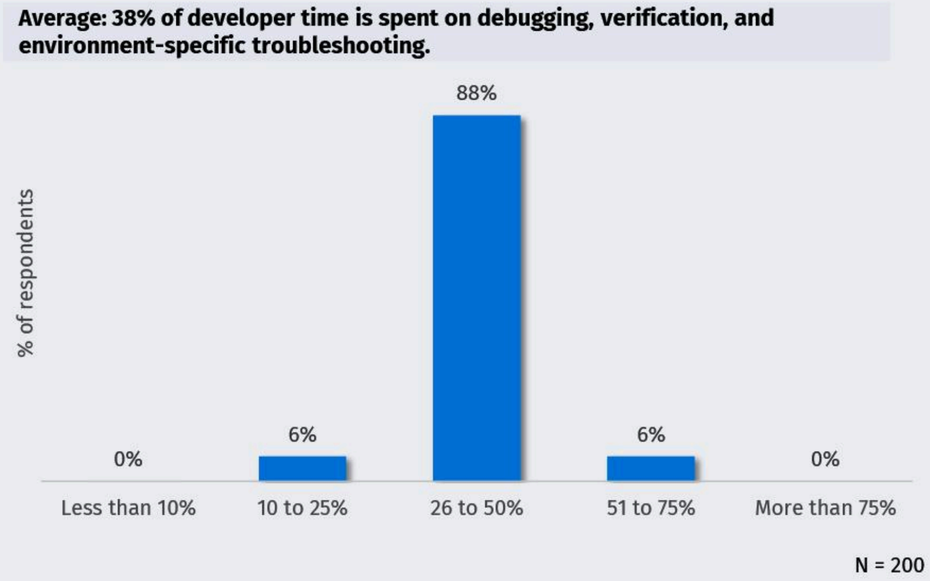


Figure 5: Proportion of Weekly Developer Time Spent on Debugging, Verification, and Environment-Specific Troubleshooting

Primary bottleneck in today's incident resolution workflows is lack of visibility in live environments

According to 60% of survey respondents, the main bottleneck in incident resolution workflows is caused by a lack of understanding of system behavior under live Production conditions (Figure 6). This reinforces the issue that AI is currently incapable of seeing how code behaves as it runs in a live environment.

Other significant bottlenecks include root cause identification (20%), the inability to act on findings due to insufficient runtime evidence (14%), which is also related to understanding system behavior in Production, and fix validation across environments (6%).

Notably, none of the respondents reported having no bottlenecks in their workflows at all, once again reflecting the limitations of AI in providing foolproof code to deal with incident resolution.

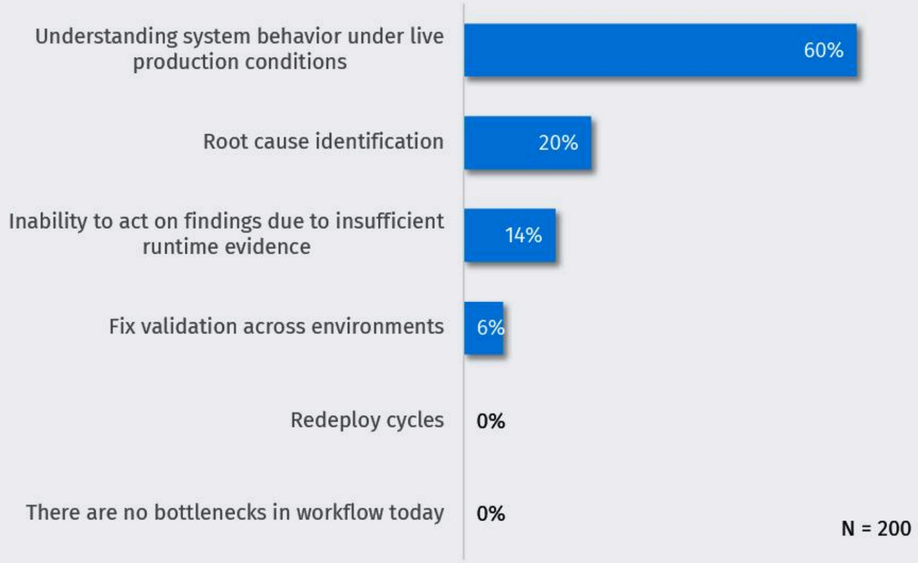


Figure 6: Primary Bottleneck in Today's Incident Resolution Workflows

High-stakes incidents demand a level of reliability AI has yet to provide

Tribal knowledge refers to internal, organizational knowledge accumulated by an engineering team, rather than data obtained from tools that provide automated triage and root-cause analysis, from AI-assisted tools.

Despite the availability of AI SREs and automated diagnostic tools, 54% of high-severity incidents are resolved by relying instead on tribal knowledge (Figure 7). This points to a widespread lack of trust in these solutions at critical moments.

Over 60% of teams fall back on tribal knowledge often (in 51-75% of high severity cases), while 37% rely on it “sometimes” (to solve 26-50% of high-severity issues).

When further investigating these results by industry, we found that reliance is most acute in the Finance industry, where 74% of respondents use tribal knowledge over automated diagnostic data during serious incidents, compared to 44% in tech (Figure 8). This indicates that in high-stakes situations, where errors are extremely costly, teams still do not trust AI to shoulder the burden.

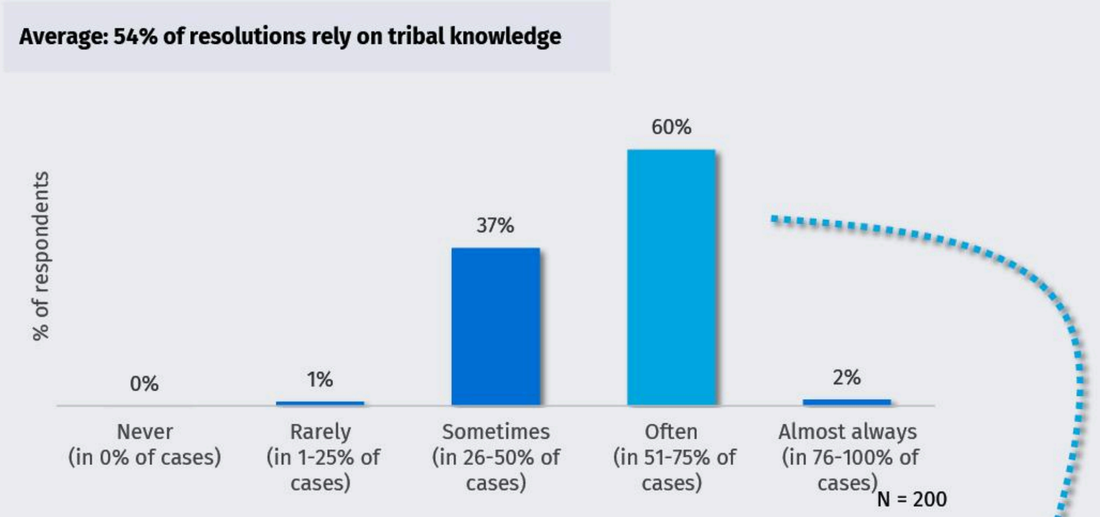


Figure 7: The Fallback to Tribal Knowledge Rather Than Relying on Documented or Automated Diagnostic Evidence to Resolve High-Severity Incidents

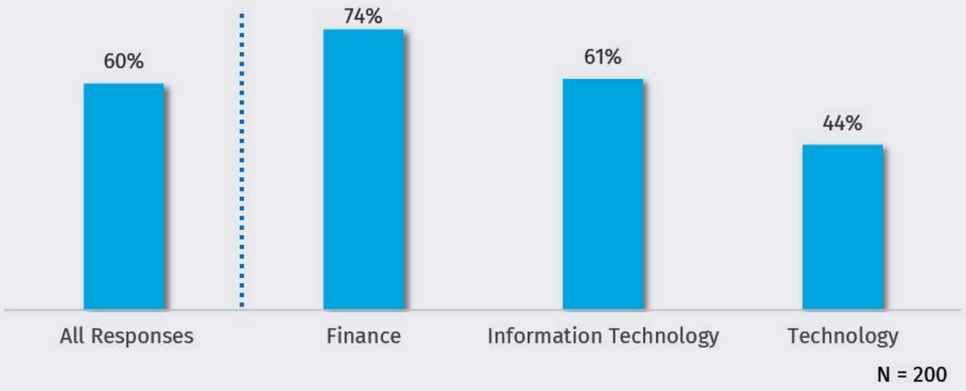


Figure 8: 51%-75% High-Severity Incident by Industry

Almost half of all escalations to SREs are for code-level issues

SREs are highly specialized engineers typically brought in only for the most complex issues that developer teams cannot solve on their own.

On average, 49% of Production incidents escalated to SREs are ultimately application-level or code-related issues, i.e., problems in how the applications were built, as opposed to infrastructure failures such as hardware breakdowns or server outages (Figure 9).

Further investigation of these results by level of seniority revealed a disconnect, with 56% of Directors reporting that SRE escalations are often code-level issues, compared to just 38% of VPs (Figure 10). This indicates that those managing the day-to-day resolution path recognize reliability as a code-level challenge even more than senior leadership driving AI adoption.

Most of the respondents (91%), in fact, indicated that this is the case for between 26-75% of Production incidents.

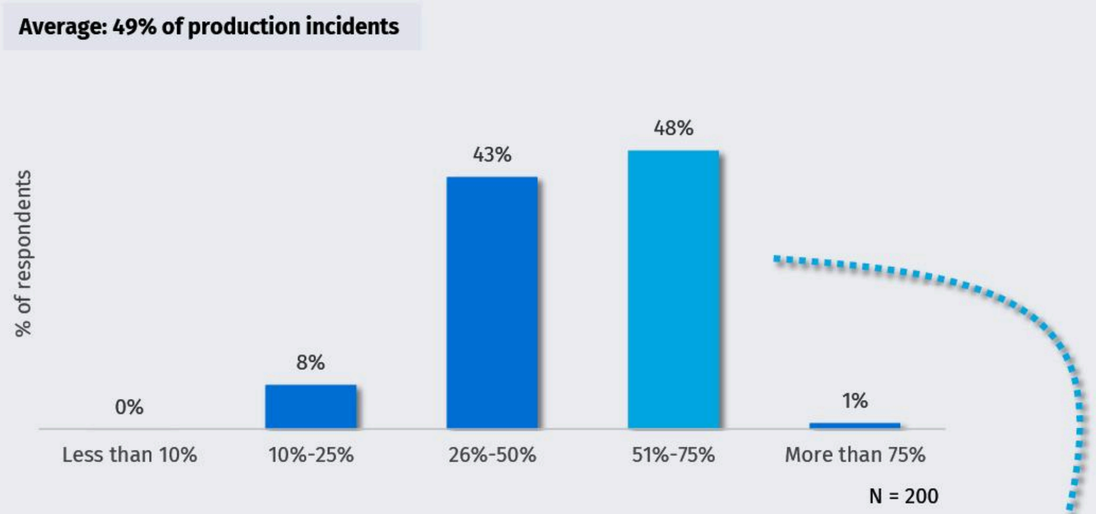


Figure 9: Incident Escalation to SREs: Application or Code-Level Issues vs. Infrastructure Failures

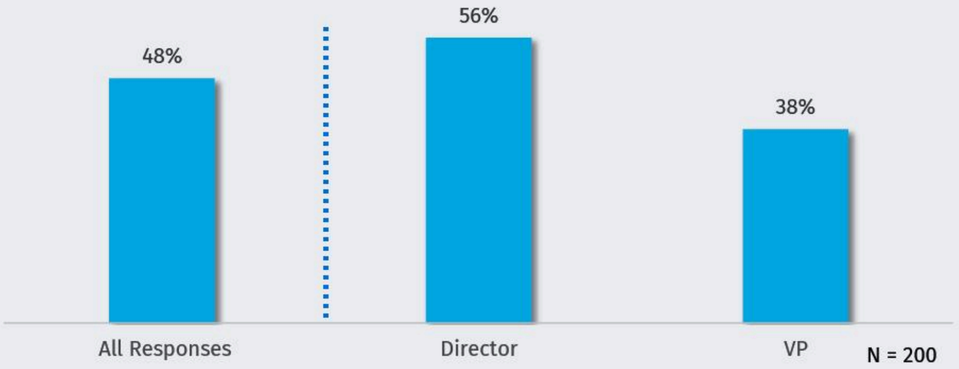


Figure 10: 51%-75% High-Severity Incident by Seniority

The Runtime Visibility Gap

AI tools are mostly blind to runtime, unable to generate their own evidence. Relying on pre-instrumented telemetry, static sources, and probability, they cannot validate root cause hypotheses or verify if proposed fixes behave correctly in Production. This reliance causes high levels of mistrust in risky environments.

AI SRE and APM investigations fail frequently due to missing execution data

AI SRE and APM investigations fail frequently due to missing execution data

AI SREs are autonomous agents that reason over telemetry and codebases to conduct root cause analysis, test hypotheses, and orchestrate incident response. AI-powered APMs provide data regarding how software performance for incident investigations and remediation.

The survey revealed that, on average, in 44% of cases where AI SREs or APM tools were used to investigate Production issues, they failed because the necessary execution-level data was never captured.

For most respondents (78%) this failure occurred sometimes (in 26-50% of cases), but for an additional 22% of respondents, investigations failed often (in 51-75% of cases) due to missing execution-level data.

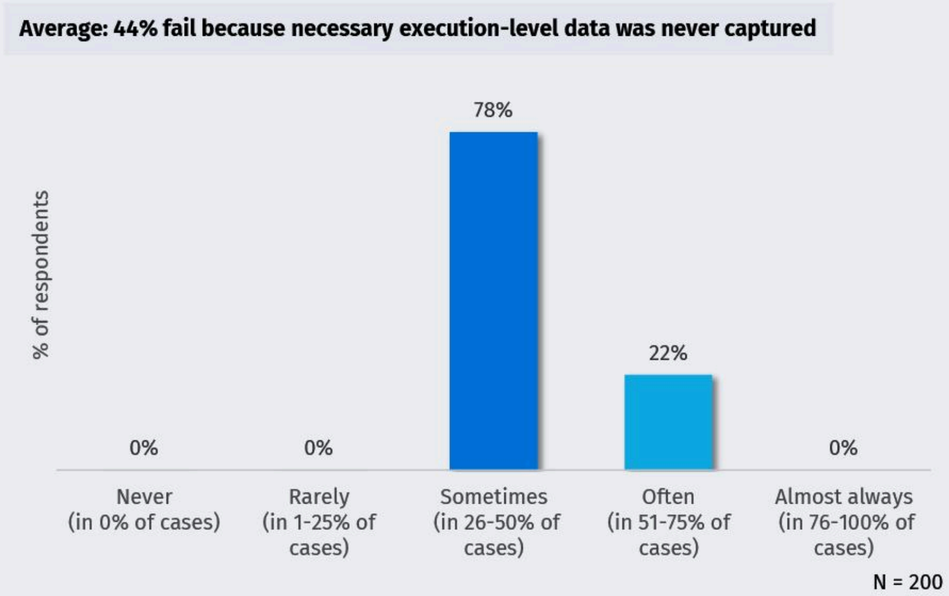


Figure 11: Frequency of AI SRE or APM Investigation Failure Due to Missing Execution-Level Data

AI agents have low levels of visibility into live execution state in Production environments

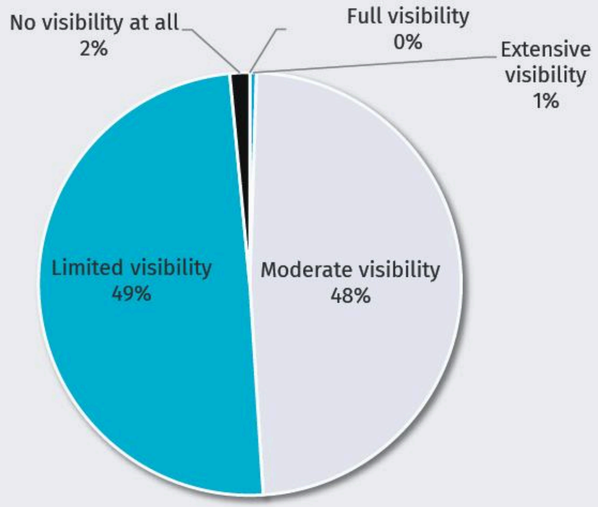
To identify the root cause of failures associated with AI agent visibility in current engineering processes (writing software code or identifying code-related issues), we analyzed the extent to which AI agents can see how live software behaves in Production.

The vast majority (97%) of leaders report significant visibility issues.

- Approximately half of companies (49%) report their AI agents have only limited visibility into live execution states, such as variables, runtime state, memory, and traffic paths.

- Only 1% of respondents report extensive visibility, and not a single respondent (0%) claimed their AI agents have full visibility.

This confirms that the lack of runtime visibility is a primary bottleneck in incident remediation and a central cause of AI SRE investigation failure. It explains the high preference among SRE and DevOps leaders to abandon automation in favor of manual validation.



N = 200

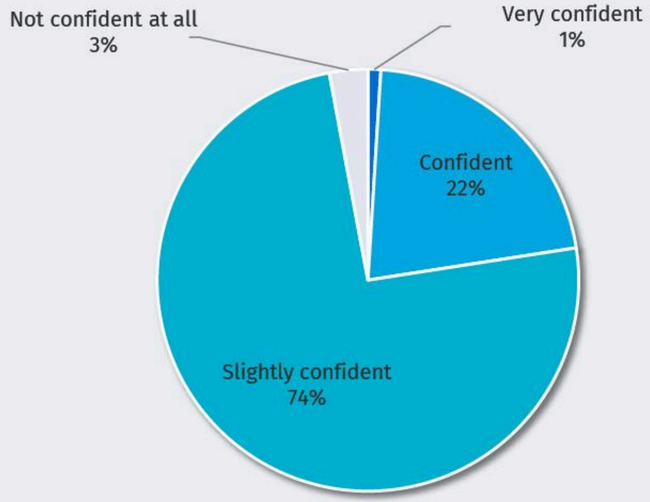
Figure 12: Level of AI Agent Visibility into Live Execution State Within Production Environments

Confidence in the current observability stack to enable AI incident remediation is low

77% of respondents stated that they have low, or no confidence that their current observability stack provides enough information for reliable, automated code-level root cause identification and remediation.

Only a minority of leaders (23%) reported that they are confident in their observability stack's ability to support this critical work.

The overall low level of confidence is a serious concern for organizations seeking to keep up the pace of reliability processes with that of AI-accelerated code generation. Without a high-confidence stack that offers runtime visibility into live execution, the transition to autonomous remediation remains blocked.



N = 200

Figure 13: Confidence in Observability Stack to Support Automated Code-Level Root Cause Analysis and Remediation

The AI SRE Trust & Adoption Gap

While organizations are interested in AI SRE and their ability to automate reliability processes, a fundamental gap exists between the category's promise and current market capabilities. To bridge this gap, AI SREs must evolve into active participants in incident investigations that find, rather than just receive, evidence across a wide array of sources.

Demand is high for AI SREs that can deliver accurate incident remediation, yet adoption is stalled

The survey revealed a significant gap between the initial evaluation of AI SRE tools and their actual integration into Production workflows.

Most enterprise respondents (90%) are currently piloting or experimenting with AI SRE tools.

The remaining 10% had evaluated such tools but chose not to adopt them at this time. Interestingly, none of the respondents were actively using these AI SREs in their production processes.

These findings are yet another example of AI hesitancy. They demonstrate that while there is a clear desire to embrace AI SRE tools, there is a gap between the “promise” of the category and the current market capabilities.

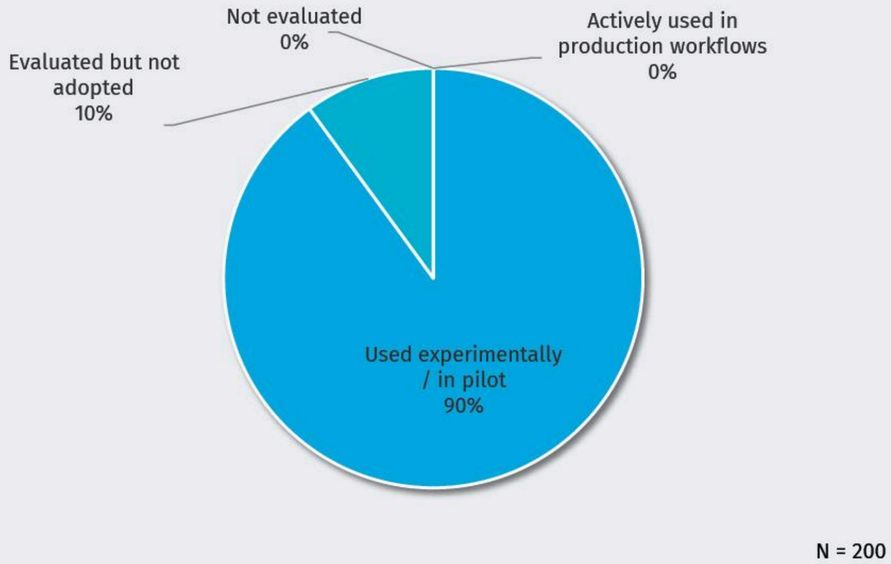


Figure 14: Current Organizational Status and Adoption of AI SRE Tools for Incident Analysis or Remediation

Engineering teams lack the trust to allow AI systems to operate in Production

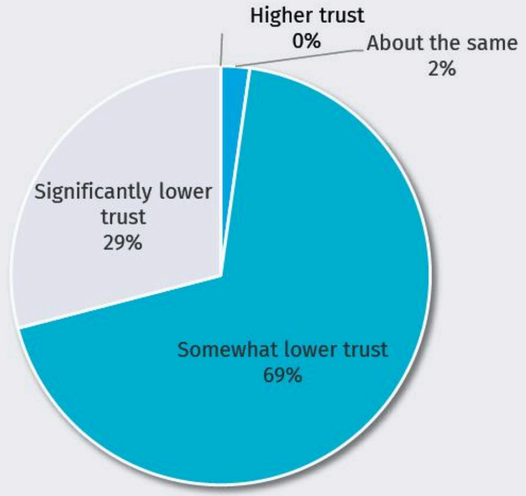
While AI coding assistants have been widely adopted, trust in AI systems operating in live Production environments remains fragile.

The survey revealed that almost all respondents (98%) who have used AI SRE tools experimentally trust them less than they trust AI coding assistants.

Nearly a third (29%) rate their trust in Production of AI as "significantly lower" than coding assistants, while 69% rate it as "somewhat lower."

Only 2% of respondents trust both AI SRE tools and AI coding assistants equally, and none trust their AI SRE tools more than their AI coding assistants.

This trust discrepancy suggests that while leadership recognizes the value of AI in code creation, the higher stakes of the Production environment demand a level of reliability current AI SREs have yet to provide.



N = 179

Figure 15: Comparative Trust in AI Coding Assistants vs. AI Systems Operating in Production

Visibility of live system behavior is the key to team’s trusting AI SREs

To move beyond experimental pilots, engineering leaders are clear on the specific capabilities required to unlock full trust and allow for the deployment of AI-suggested fixes.

The top requirement for 58% of SREs and DevOps leaders is the ability for the tool to generate 'evidence traces' of variables at the point of failure around the incident site.

The second most chosen capability, cited by 42% of respondents, is for these AI tools to reliably verify suggested fixes before they are deployed.

While the ability to ingest multiple sources of logs or to provide better natural language explanations were also offered as potential ‘top capabilities’, no respondents chose these options as critical for establishing trust.

The fact that all responses coalesce around live system behavior indicates that runtime verification is the absolute priority for SRE and DevOps leaders.

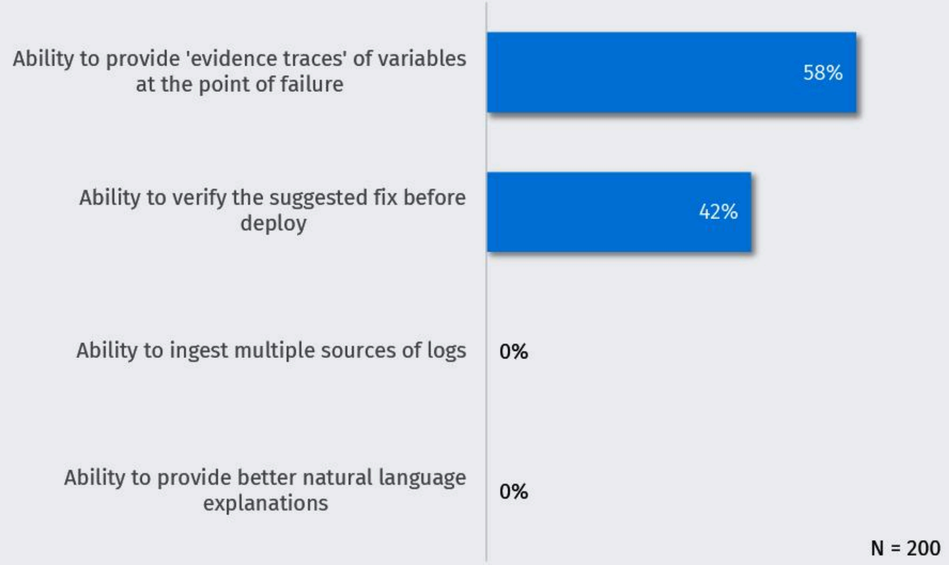


Figure 16: Most Important Capability for Trusting AI SRE Incident Remediation Proposed

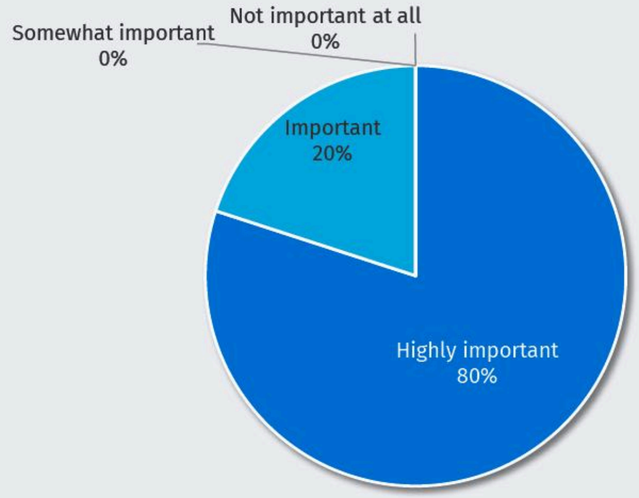
Full-stack visibility is mandatory to bridge the trust gap and enable successful AI adoption

AI SRE tools are being developed by companies of all sizes, from startups to Fortune 500s.

The largest companies typically offer AI SREs that are part of their closed corporate ecosystem, accessing datasets generated from their sister products. However, other companies have developed similar AI SREs that are platform agnostic and can be integrated with any data source that they are connected to, to run their reasoning processes over.

The survey revealed that all respondents believe that it is essential that AI SRE tools can access and reason over data from all data sources and environments in their stack. Most respondents (80%) rated this capability as "highly important," while 20% rated it as "important".

The consensus among engineering leaders is that AI SRE tools constrained to a single observability stack or vendor fail in real-world Production settings.



N = 200

Figure 17: Importance of AI SRE Access to All Tools, Data Sources, and Environments Over Single-Vendor Data Analysis

Report demographics

Country, Industry, Company Size, Job Seniority, Role and Current Role Experience

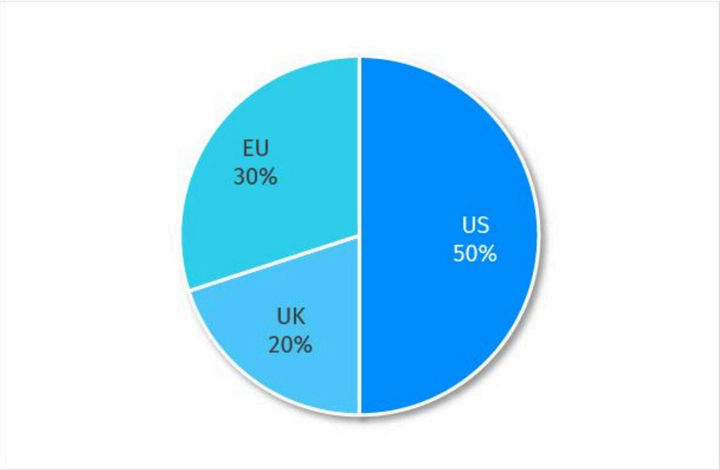


Figure 18: Country

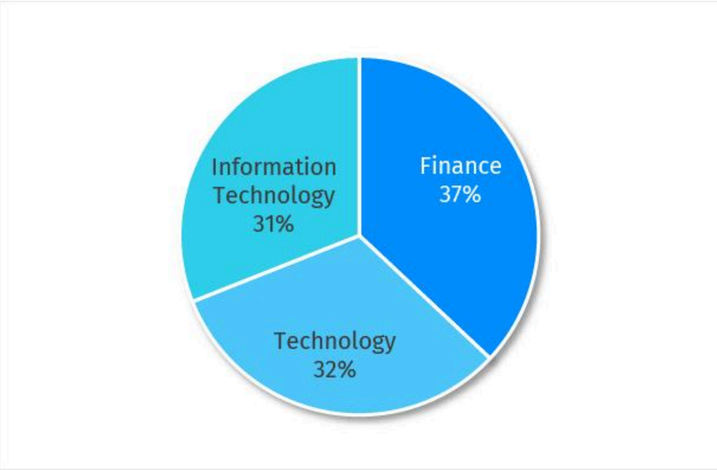


Figure 19: Industry

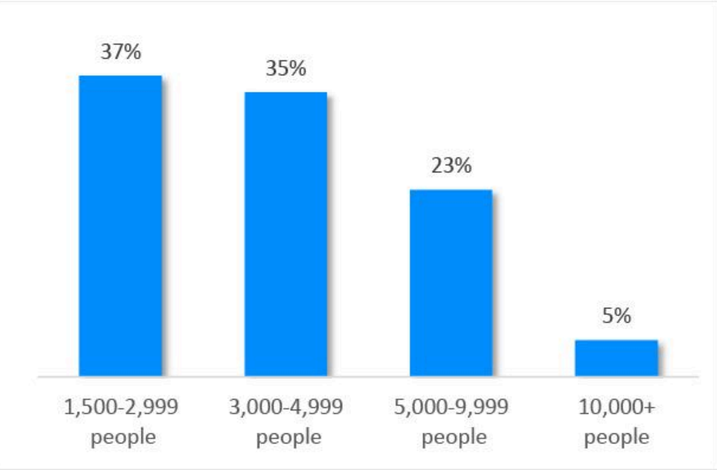


Figure 20: Company Size

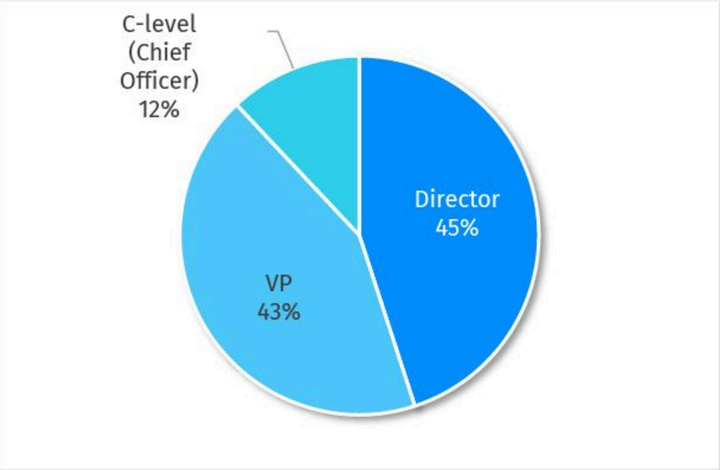


Figure 21: Job Seniority

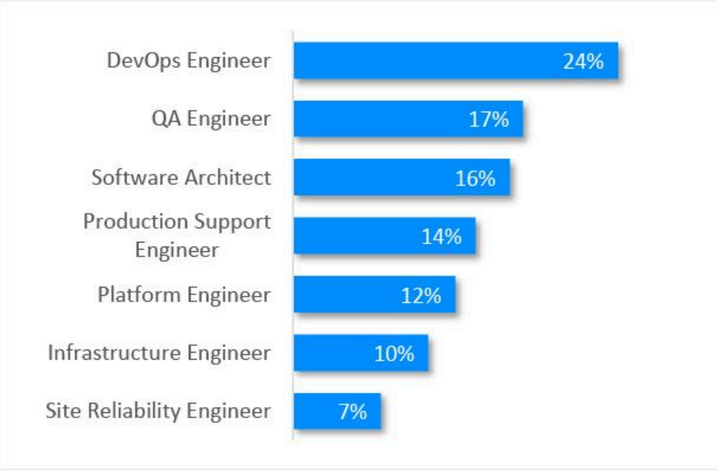


Figure 22: Role

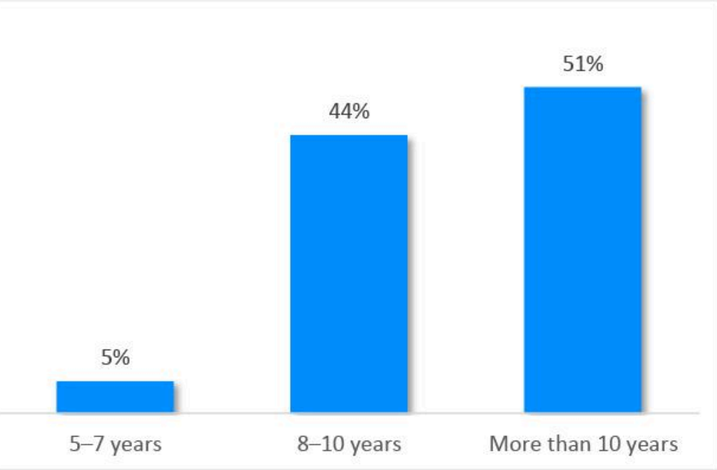


Figure 23: Current Role Experience

About Lightrun

Lightrun is an AI-native reliability engineering platform that enables engineering teams and AI agents to autonomously prevent and remediate software issues across the SDLC, from code to Production.

As AI workloads and AI-generated code accelerate, developers and SREs are spending more time validating how software behaves in real runtime environments, where new “unknown unknowns” emerge. Lightrun’s end-to-end autonomous AI SRE operates from the moment an issue arises through root cause analysis to validated fix — generating telemetry at the line level to see code in action in real time, understand its behavior in live environments, prove RCA against live execution, and validate fixes directly in runtime without redeployments.

Trusted by global enterprises such as AT&T, Citi, Microsoft, Salesforce, UnitedHealth Group, SAP, ICE/NYSE, ADP, HPE, and Booking Holdings, Lightrun increases developer productivity, reduces business and compliance risk, and cuts MTTR to minutes. The company is backed by $110M in funding from leading investors including Accel and Insight Partners.

Book a demo

Visit us:
www.lightrun.com