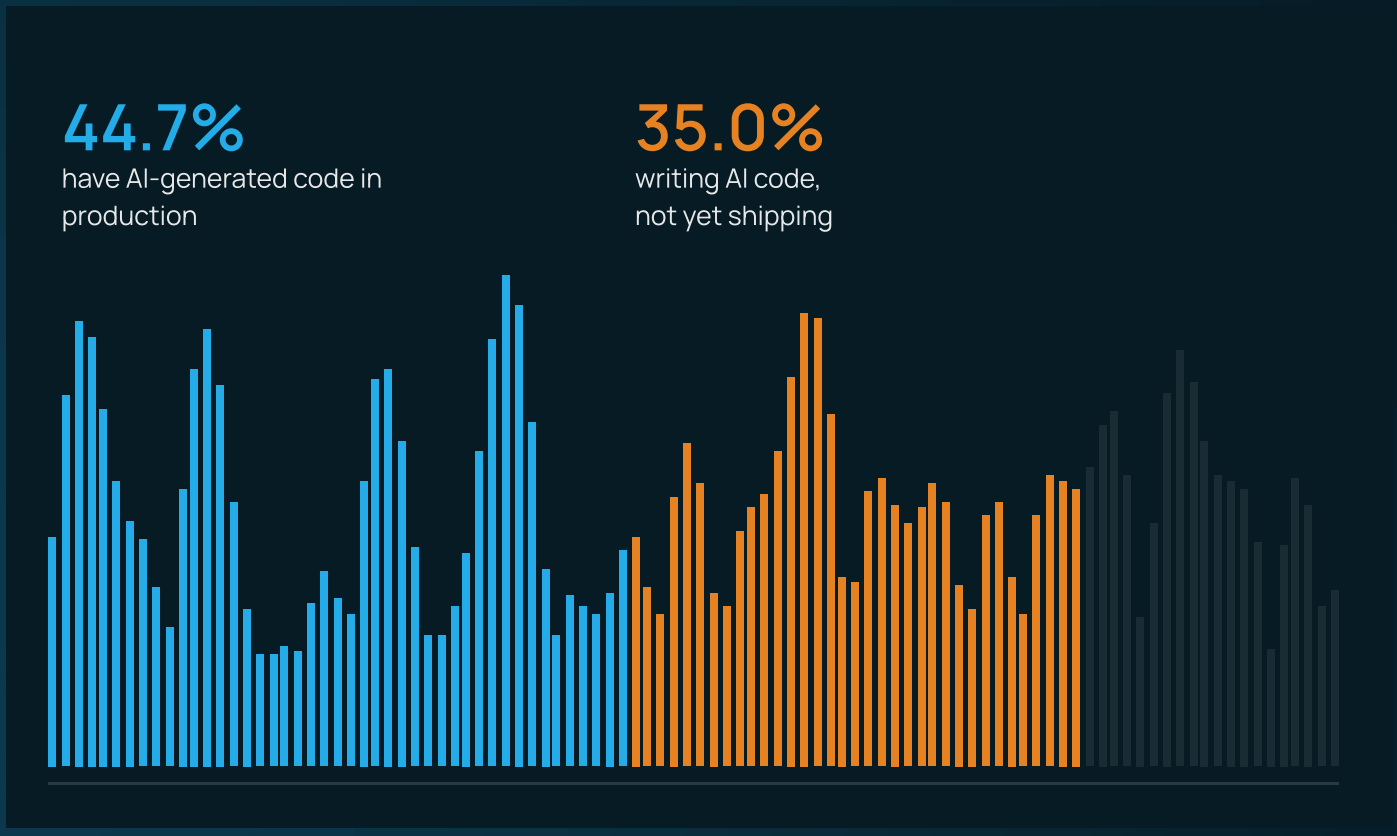
REPORT

AI code generation reality check

What engineering leaders need to know



FLUX

RESEARCH BY DIMENSIONAL RESEARCH
SPONSORED BY FLUX

OVERVIEW

Executive summary

AI code generation has crossed the hype phase and become table stakes, but most organizations are still pretending the old ways of managing code will scale. Leaders celebrate productivity gains while flying blind on what's changing in their codebases, and the result is growing invisible risk, compounding technical debt, and brittle systems no one fully understands.

This survey of 309 engineering leaders and practitioners shows a simple, uncomfortable truth: AI has solved the code creation problem faster than organizations have solved the comprehension and control problem in their codebases. Nearly every company now uses AI to write code, but a large share is still hesitant to ship that code to production, because they do not trust their ability to see and manage the risks.

The AI Code Generation Reality Check report examines how AI-generated code is changing work in enterprise engineering organizations. Nearly all respondents are directly involved in code review: 92.6% manage people who review code, and 59.2% personally perform code review, so the findings reflect both hands-on experience with AI-generated code and executive-level perspectives. We asked how they manage AI-generated code, how it affects code quality and security, and which strategies work for balancing speed with reliability.

This survey focuses on three key areas:

Leadership and expectations

What AI-accelerated development really looks like for executives and engineering leadership, and where expectations and quality differ from reality.

Team management and productivity

How AI changes code output, review load, and visibility into what's changing in the codebase, creating both efficiency gains and new points of friction.

Code quality and risk management

How organizations are responding to emerging quality, security, and technical debt risks from AI-generated code, and which safeguards they are investing in.

Productivity gains are real, but so are the risks. With most respondents already changing their development and release processes to accommodate AI-generated code, and many still seeing significant negative impacts, it's clear that best practices are still emerging.

CORE INSIGHTS

Key findings

The AI Code Generation Reality Check survey was conducted independently to provide engineering leaders with objective insights into AI code generation practices today. This report is based on research conducted by Dimensional Research, surveying 309 engineering leaders and practitioners at organizations that are already using, piloting, or planning to use AI-generated code. Respondents with no plans to adopt AI-generated code or who were unsure were excluded from the analysis. See Appendix A and B for the complete methodology and respondent profile.

Almost everyone uses AI to write code, but many don't ship it.

44.7% of companies have AI-generated code in production. Another 35.0% use AI for coding but haven't shipped it. In other words, many organizations already trust AI to write code but aren't yet comfortable enough to put it in front of customers.

AI-generated code usage is concentrated in a few low-risk areas.

Teams report gains in documentation (68%), unit testing (66%), and simple functions (58%). This shows that organizations start using AI where patterns are repetitive and errors are lower risk.

AI productivity gains are outpacing review and risk capacity.

80.5% of organizations have already changed their development and release processes for AI-generated code, yet many still struggle to understand what's changing and to detect security, dependency, and performance issues week to week.

AI-generated code has become an enterprise-wide quality and risk concern.

Stakeholders across security (62.5%), compliance (51.5%), CTO/CIO leadership (46.9%), and legal (40.8%) report concerns, and organizations are seeing concrete negative impacts, even as they continue to adopt AI-generated code for its productivity and cost benefits.

Safeguards are becoming standard production tooling.

46% have bought code quality analysis tools and 39% have added automated review solutions, while 76% say a solution that mitigates the risks of AI-generated code would be very or extremely valuable. In addition, 65% believe AI could be better than humans at code review, signaling that organizations are betting on AI-powered safeguards to manage the risks AI creates.

INTRODUCTION

AI code generation is working; are your practices keeping up?

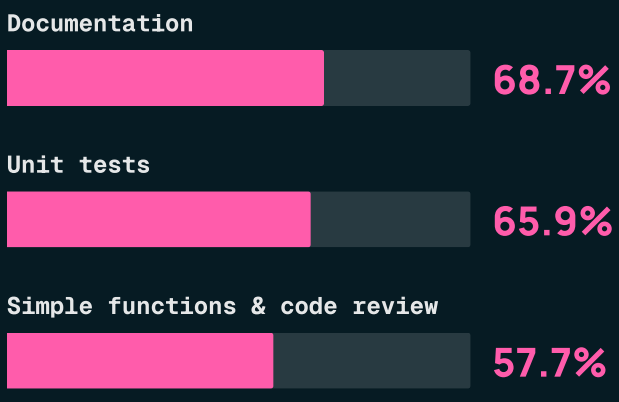
Engineering teams are shipping more code than ever, often with the same review capacity they had years ago. Leaders see impressive productivity gains on paper, but behind the scenes they're wrestling with weirder code, faster-moving changes, and growing blind spots in quality, security, and technical debt.

AI was supposed to make development simpler. Instead, it has shifted the bottleneck from writing code to understanding and trusting it. What the data shows is not a simple success story or a horror story, but a mixed reality: real gains, real issues, and a widening gap between coding speed and review, testing, and risk management.

Flux sponsored this independent research with Dimensional Research to get an accurate picture of where AI-generated code stands today: how widely it's used, where it's breaking down in practice, which risks matter most, and what high-performing organizations are doing differently to keep pace without losing control.

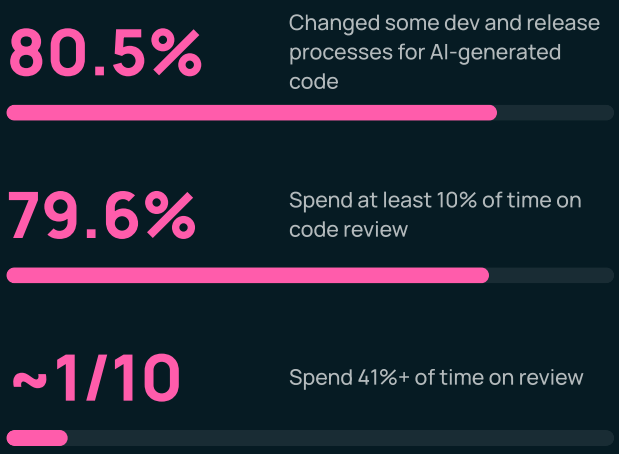
AI use highest in low-risk, repetitive work.

Organizations start where patterns are clear and errors are easier to contain.



Productivity is up. Review capacity isn't.

AI is generating more code than teams can comfortably review and validate.



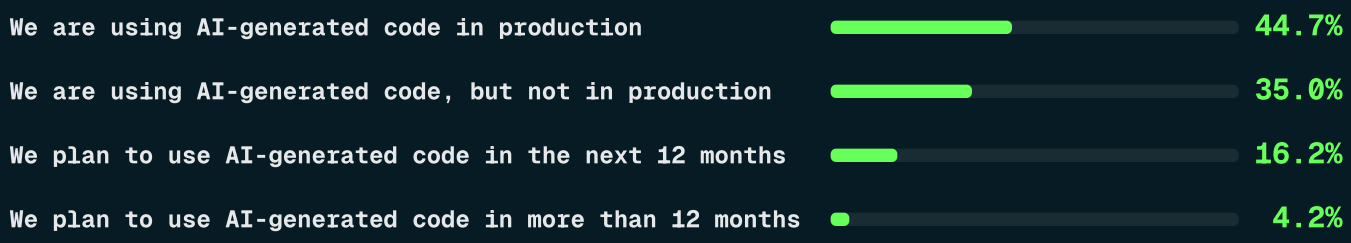
FINDING 1

Adoption for AI coding is high, but trust in deployment still lags

The conversation has shifted from whether to use AI-generated code to how to deploy it safely at scale. At the same time, most engineering leaders still rely on ticket-based and human-reported tools built for a human-speed world. These tools were never designed for AI-assisted development, where agents and coding copilots have increased code velocity and volume by orders of magnitude.

AI code adoption status

Respondents shared whether their company is using AI-generated code in the software development process.



The real question isn't whether to use AI to write code; it's how much AI-generated code you'll put in front of customers and how clearly you can see the changes in your codebase.

The numbers confirm the adoption of AI code generation:

44.7%
of companies already have AI-generated code in production.

35.0%
are actively coding with AI, though not yet deploying to production.

< 5%
report no plans to use AI-generated code in the next year.

This shows near-universal adoption. But deployment rates tell a different story: over a third of the organizations using AI to write code haven't pushed it to production yet. They're stuck between wanting the speed gains and worrying about what will break when they ship.

That gap between coding with AI and deploying AI-generated code reveals the real challenge. Writing code with AI is easy, but trusting it enough to put it in front of customers is much harder.

Can AI fix the problems it creates?

Faster code generation creates a different problem: finding time to review all that newly generated code. Code review is time-consuming, too. Some engineering leaders think AI can help review the code it generates, but the question isn't just whether AI can review code, it's whether AI can match or beat human performance at catching the problems that matter. When we asked whether AI could be better at reviewing code than software developers, nearly two-thirds said yes: 64.9% believe AI could outperform humans in at least some aspects of review, while only 21.1% disagreed and 14.0% had no opinion.

Can AI outperform humans at code review?

Respondents shared whether they believed AI could be better at code review than software developers.

Respondents also see AI's strengths in code review lining up with the kinds of work humans struggle to sustain at scale.

- **57.4%** said AI could apply uniform standards more consistently and provide more thorough analysis
- **53.3%** believed AI could better identify patterns and connections across different codebases
- **51.8%** anticipated quicker feedback
- **49.7%** indicated AI could detect more issues
- **48.2%** emphasized that it provides 24/7 availability for code review

Where AI may exceed human code reviewers

Respondents indicated that AI may be better at reviewing code than software developers in these areas.

What this means for leaders: Treat the decision to ship AI-generated code as a risk decision, not just a tooling decision, and ask explicitly what proof you have that current safeguards match where AI is already writing code.

FINDING 2

Trust in AI-generated code is high in low-risk, repetitive tasks

AI-generated code is certainly delivering value, but the question is where? We asked organizations which tasks they're using AI for, and the results show a clear pattern.

Organizations are already seeing tangible benefits from AI in:

Documentation (68%), streamlining one of the most time-consuming aspects of development.

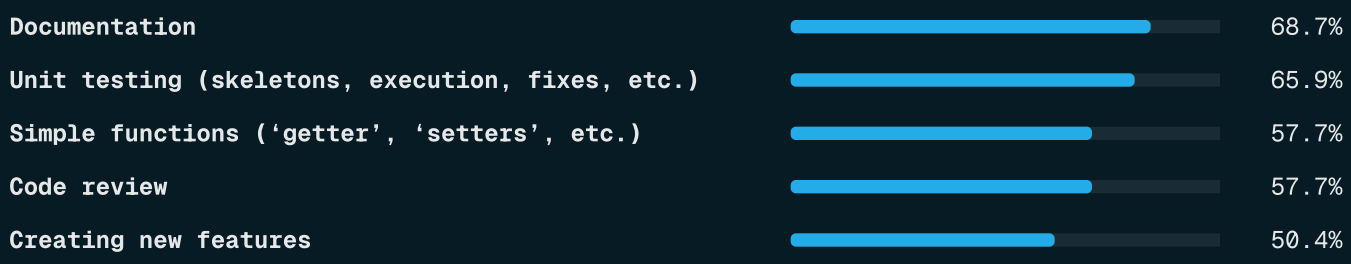
Unit testing (66%), improving both speed and coverage.

Simple functions and code review (58%), accelerating routine work and freeing humans to focus on harder problems.

Starting with low-risk, repetitive work is only half the story; the other half is whether AI is delivering on the productivity, cost, and quality gains leaders are counting on.

Tasks where AI-generated code is being used

Respondents shared what types of tasks AI-generated code is being used for.



On speed and cost, expectations and reality are directionally aligned.

68.3% of respondents who have not yet adopted AI-generated code expect increased productivity

54% improved code efficiency

52.4% faster prototyping

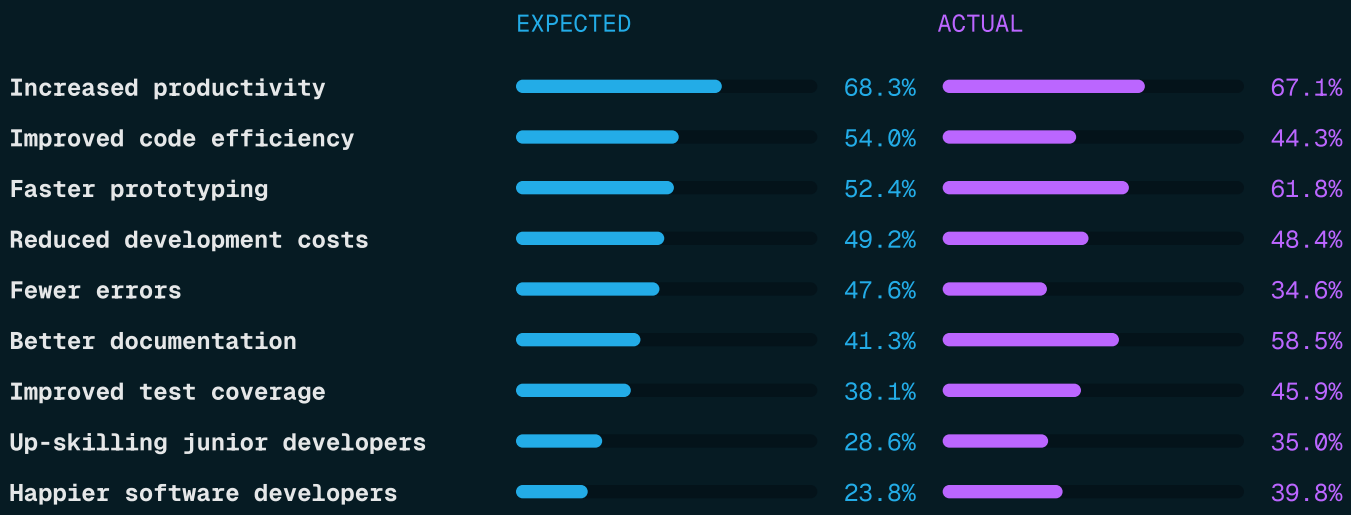
49.2% reduced development costs

Among current users, 67.1% report increased productivity, 61.8% faster prototyping, 58.5% better documentation, and 48.4% report reduced development costs, almost exactly matching those expectations.

Quality is where the gap opens up. While 47.6% of non-users expect AI-generated code to reduce errors, only 34.6% of current users say they are actually seeing fewer errors—a double-digit shortfall that shows quality gains are proving harder to realize than productivity or cost improvements.

Expected vs. actual benefits of AI-generated code

Left: what non-adopters expect. Right: what current users report experiencing.



Taken together, these results show that organizations are putting AI to work first where tasks are repetitive, patterns are clear, and failures are easier to contain before trusting it with higher-impact changes. Expectations and reality largely match on productivity and cost, but not yet on error reduction.

What this means for leaders: Treat AI-generated code as a productivity accelerator, not a guarantee, and put explicit guardrails in place (strong reviews, tests, and code-quality monitoring) before you expand it into higher-impact changes.

FINDING 3

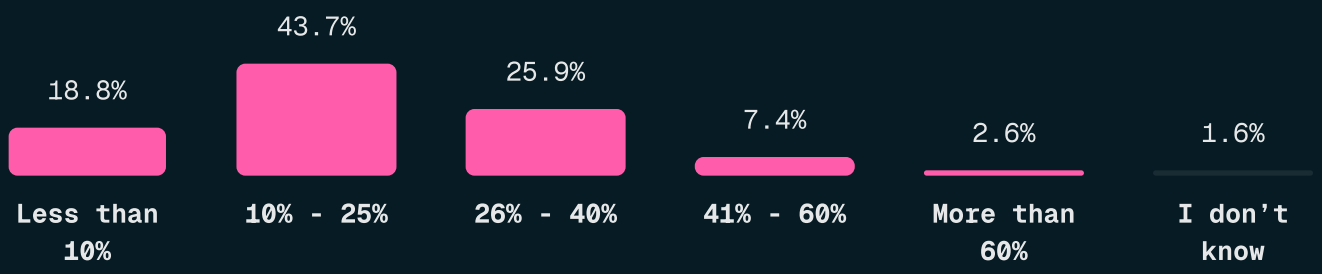
AI productivity gains create an AI code evaluation and visibility gap

Productivity gains from AI-generated code don't come without cost. For many developers, code review already consumes a significant share of the week; nearly 80% of respondents spend at least 10% of their time on review, and about one in ten spend 41% or more, so AI-generated code adds to an already heavy load.

As teams generate more code, they also create more to review, more to maintain, and more places for bugs to hide. This creates a new gap between how fast teams can generate code and how confidently they can validate it. What used to be a relatively predictable code review process is becoming harder to manage as pull requests change in size, frequency, and composition.

Time spent on code review

Respondents shared approximately how much time is typically consumed by code review.



Generated code risks upsetting the balance teams had built over years. Reviewers no longer know exactly what to expect from a change, especially as tools evolve, teams experiment with new workflows, and more non-engineers get involved in code generation. The good news is that teams can generate more code; the bad news is that the resulting code often looks different than before, making reviews slower and less reliable.

This shift creates an AI visibility gap: leaders find it harder to see and understand what is changing in the codebase as AI increases code volume. When code volume rises, small but critical modifications are more likely to be buried in noise, and security issues or breaking changes are more likely to slip through to production.

Visibility into code changes is partial at best

When we asked about the biggest challenges in understanding changes in the codebase, four themes stood out. Over half (53.7%) of respondents pointed to complex code as a primary obstacle, followed by 46.3% who cited different development teams with different approaches. 43.0% highlighted poor documentation, and 37.9% called out the large volume of changes as a major challenge in itself. Other challenges, such as new dependencies (35.0%), strategic architecture changes (33.3%), developer turnover (32.7%), and unhelpful commit messages (25.6%), were also common.

Biggest challenges understanding codebase changes

Respondents shared their biggest challenges in understanding changes in their codebase.

How AI-generated code is changing work on the ground

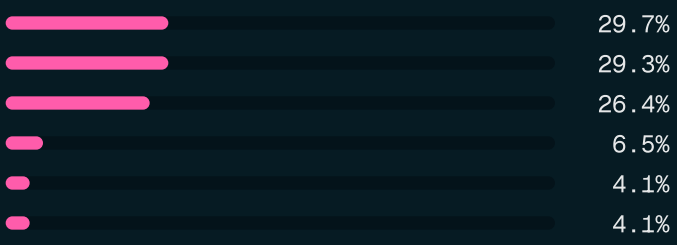
To understand how AI-generated code feels day-to-day, we asked about issues, negative impacts, visibility, and process change.

How issues from AI-generated code compare to human-generated code.

When we asked how the number of issues from AI-generated code compares to human-generated code, about a third of respondents (32.9%) said AI-generated code creates somewhat or substantially more issues, 33.4% said it creates somewhat or substantially fewer issues, and 29.7% said it is about the same.

Issues from AI-generated vs. human-generated code

Respondents shared how issues from AI-generated code compare to human-generated code.



Negative impacts organizations report from AI-generated code.

When asked about negative impacts from AI-generated code, respondents most often pointed to reduced learning opportunities for junior developers (just over 40%), along with code that doesn't deliver the requested functionality, unintended dependencies, security vulnerabilities, and hard-to-debug changes (each reported by roughly 30% of organizations). Other issues were also common (unmaintainable code, performance problems, compliance violations, changing code architecture, difficult-to-customize or inefficient code, and lack of documentation) while only a small minority (6.3%) said they have not experienced negative impacts from AI-generated code.

Negative impacts organizations report from AI-generated code

Respondents shared negative impacts from AI-generated code at their organization.

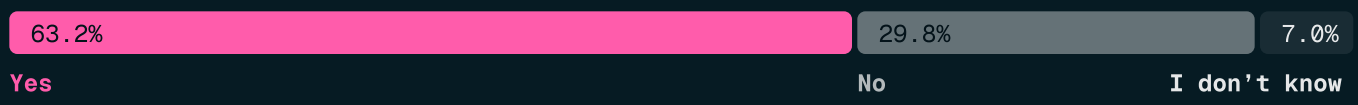
AI-generated code is not universally better or worse, but it is undeniably different.

Leadership visibility into AI-generated changes

63% of respondents say their management can stay on top of week-to-week changes across all code, but nearly a third say they cannot. Our data on difficult-to-detect changes suggests that even the confident group may be missing more than they think.

Can leadership stay on top of week-to-week code changes?

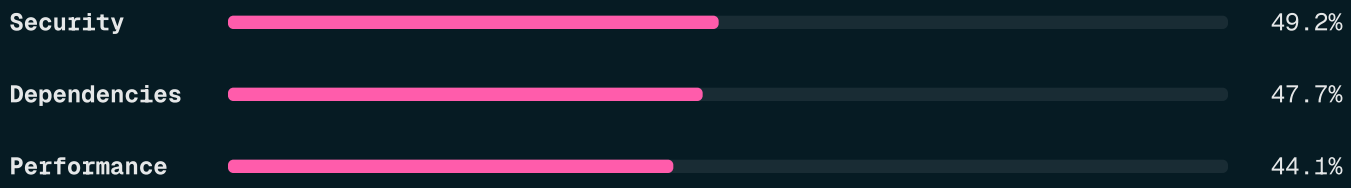
Respondents shared whether their management can stay on top of week-to-week changes across all code.



When we asked which week-to-week changes are hardest to detect, respondents pointed first to security issues (49.2%), dependency changes (47.7%), and performance impacts (44.1%). Core business logic (33.8%), integrations with external websites (33.8%), infrastructure (29.7%), and database changes (22.6%) also show up as significant blind spots. These are exactly the kinds of changes that cause costly, high-severity incidents when they go unnoticed. Only a small minority said they don't attempt to evaluate week-to-week changes (4.1%) or that none of these changes are difficult to detect (2.6%).

Changes hardest to detect week to week

Respondents shared which changes are difficult to detect when evaluating week-to-week code versions.



Organizations that focus on lines of code shipped, but don't track review capacity and change visibility, are setting themselves up for problems they won't see until a critical production incident forces their attention. Closing this gap requires ground-truth engineering intelligence: continuous signals from the codebase itself about what is changing, where complexity is growing, and where review capacity is falling behind.

AI was supposed to make development simpler. Instead, it has shifted the bottleneck from writing code to understanding and trusting it.

Process changes driven by AI-generated code

Most organizations are already changing how they work to accommodate AI-generated code. 80.5% reported that AI-generated code required changes to their software development and release processes (15.9% major, 64.6% minor), while only 18.3% reported no changes. The most common responses include new policies for AI use, additional training, more robust code reviews, added reviewer capacity, and stricter security audits, highlighting that AI-generated code is reshaping day-to-day workflows, not just tools.

Organizational Responses to AI Code Generation

Changes to software development and release processes required by AI-generated code.

What this means for leaders: Stop measuring success only by how much code ships; start tracking whether your review capacity and change visibility can keep pace with the volume and complexity of AI-generated changes.

FINDING 4

AI-generated code quality is now an enterprise-wide quality and risk concern

Increasingly, organizations are willing to spend on safeguards for AI-generated code, and the data shows they see it as a significant source of quality, security, and compliance risk, not just another development tool.

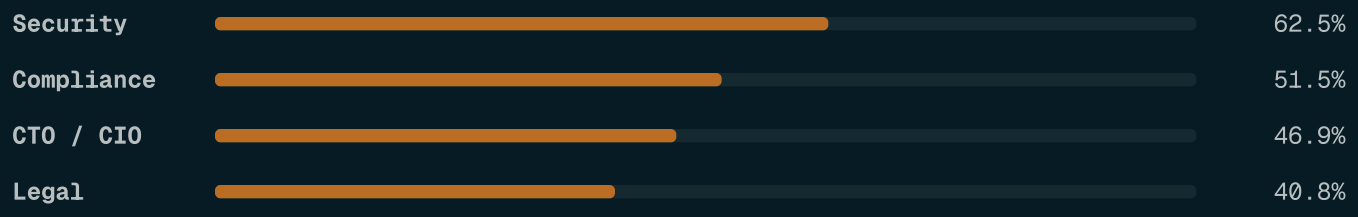
Leaders already cite a complex codebase (46.0%), simply knowing what has changed (45.0%), and integrations across systems (41.7%) as top challenges to maintaining an evolving codebase (ensuring quality, security, and compliance) with poor documentation (35.6%) close behind. Against that backdrop, when we asked which stakeholders have concerns about AI-generated code, the responses show this isn't just an engineering problem.

AI-generated code is becoming standard; what now differentiates teams is how clearly they define who owns its behavior in production and how they coordinate that across functions.

The concerns about AI-generated code span departments. Engineering leaders struggle to see emerging risks in the codebase before they hit production. Security teams are the most likely to raise flags, with nearly two-thirds of respondents saying security stakeholders are concerned about AI-generated code. Over half of compliance teams share that concern, CTO/CIO leadership are close behind, while legal teams also show significant concern at around 40%. Other stakeholders are also frequently worried, including operations (32.7%), QA (32.0%), product management (26.9%), the CEO (21.0%), customer success (14.6%), and even marketing (8.1%), underscoring that AI-generated code has become an enterprise-wide risk topic, not just an engineering detail.

Stakeholders concerned about AI-generated code

Respondents shared which stakeholders have concerns about AI-generated code at their company.



What this means for leaders: Treat AI-generated code as an enterprise risk topic, and align security, compliance, legal, and engineering on a shared view of where AI is used, what can go wrong, and who owns each class of risk.

FINDING 5

Organizations are making AI safeguards standard for production code

Companies are spending real money to address these concerns. The data suggests that organizations view risks related to AI-generated code as urgent, not experimental. When we asked what solutions organizations have purchased to manage these risks, the responses showed substantial investments in categories of tooling that barely existed two years ago.

As AI code generation adoption grows, enterprises are investing heavily in mitigating risks.

45.6% have invested in code quality analysis solutions.

39.0% are using automated code review tools

38.5% have added static application security testing (SAST)

35.9% also adopted software composition analysis (SCA)

32.3% are using interactive application security testing (IAST)

31.3% implemented training classes for specific coding solutions (assistants)

Solutions purchased to mitigate AI code risks

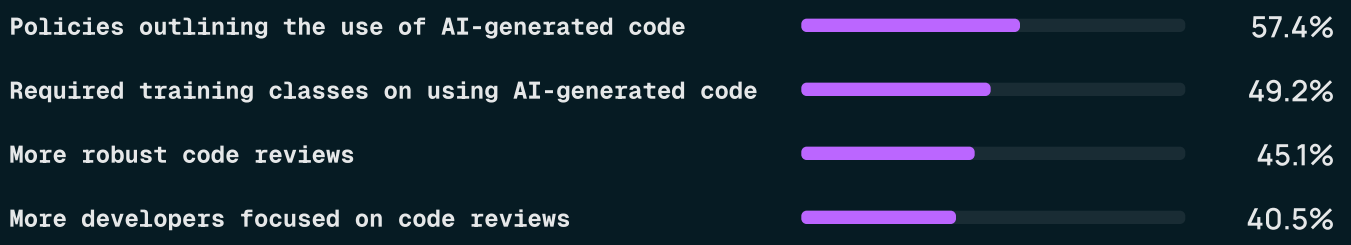
Respondents shared which new solutions their companies have purchased to mitigate risks associated with AI-generated code.

These investments reflect a shift from treating AI-generated code as an experiment to treating it as a first-class source of production risk that requires broader visibility into code quality and earlier detection of hotspots, dependencies, and brittle modules. When AI-generated code is deployed without careful review, issues are getting released to production environments. 76% of respondents said a solution that mitigates the risks of AI-generated code would be very or extremely valuable.

Beyond tooling, organizations are also changing how they work. The most common shifts include new policies outlining the use of AI-generated code (nearly 60%), required training on using AI-generated code (about half), more robust code reviews (just under half), and assigning more developers to focus on reviews (around 40%), with many also adding stricter security audits and purchasing additional safeguards such as code quality and automated review tools. Only a small minority report making no changes at all.

Process changes made to mitigate AI code risks

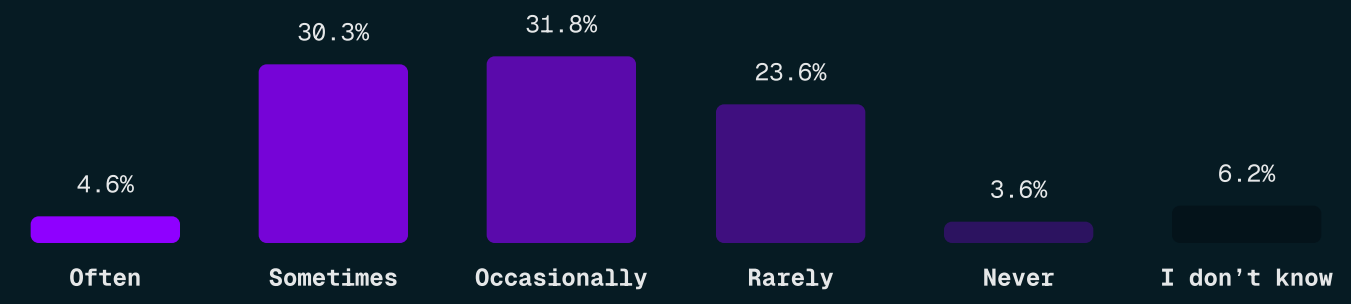
Respondents shared which changes their company made to mitigate risks with AI-generated code.



These changes are grounded in experience. Across companies already using AI-generated code, production impact is common rather than rare. Only 3.6% of respondents say AI-introduced issues never reach production, while a combined 35.0% report they do so often or sometimes and 31.8% say they occur occasionally. The remaining 23.6% say such issues happen rarely, underscoring that for most organizations, AI-originated problems in production are a known, recurring risk.

How often do AI-introduced issues reach production?

Respondents shared how frequently issues introduced by AI-generated code make it into production at their companies.



What this means for leaders: Tie every AI safeguard to hard signals from your codebase so you can see where AI is creating risk, where it's paying off, and which changes you can safely ignore.

CONCLUSION

The code isn't the hard part. The visibility and control are.

The survey shows organizations caught between real productivity gains and real costs. Nearly half of respondents have deployed AI-generated code to production, and another third use AI to write code but have yet to ship it. The review and risk bottlenecks are getting worse; what started as a developer tool has become an enterprise-wide concern for security, legal, and compliance teams.

The paradox is that AI is helping teams move faster while also creating code paths they can't fully see or control. AI-generated code is contributing to production issues, although many respondents believe AI could eventually outperform humans at code review. Teams are betting that more AI can fix the problems AI introduced, but that bet only pays off if they can consistently see what changed, who changed it, and how it affects both risk and innovation in their codebases.

What high-performing engineering teams are doing differently:

A governance decision

They treat shipping AI-generated code as a governance decision, not just an engineering decision.

Invest in safeguards

They invest in safeguards: quality analysis, automated review, and security testing, then prove those safeguards are reducing incidents and review load.

Expand review capacity

They expand review capacity and practices to match AI-accelerated output, instead of assuming existing processes will scale unchanged.

Close the visibility gap

They close the AI visibility gap with code-first intelligence that surfaces changes, dependencies, and hotspots before incidents occur, without adding more manual reporting or ticket hygiene.

Humans in the loop

They keep humans in the loop on the changes that matter most, using AI to surface the right work rather than bypass judgment.

The organizations building these capabilities now will be the ones that can let AI write more code, ship faster, and still trust every change that reaches production.

ABOUT

About Dimensional Research

Dimensional Research® provides practical market research for technology companies. We partner with our clients to deliver actionable information that reduces risks, increases customer satisfaction, and grows the business. Our researchers are experts in the applications, devices, and infrastructure used by modern businesses and their customers.

For more information, visit www.dimensionalresearch.com.

About Flux

Flux is a code-first engineering intelligence platform that helps engineering leaders make better decisions with ground-truth visibility into the work actually happening across today's complex, AI-accelerated codebases. Instead of relying on tickets, Flux reveals the work teams are doing, surfaces risk and technical debt, and connects engineering activity to business outcomes. With this visibility, leaders can distinguish innovation from maintenance, spot emerging issues before they become incidents, and understand collaboration patterns to build healthier, higher-performing teams across their engineering organization.

Learn more at www.askflux.ai, explore the resources, and follow Flux on LinkedIn.

APPENDIX A — SURVEY OVERVIEW

Research goals

The primary goal was to understand the utilization, benefits and challenges of AI generated code. The research delved into AI generated code concerns such as compliance, security, data protection and overall code quality. The research also investigated the value to the business for AI-enabled automated code review, such as time savings, freed up resources, reduced risk, and improved quality.

Methodology

Executives, and software practitioners at small to global enterprise companies were invited to participate in a survey on their company's software practices and AI utilization. The survey was administered electronically, and participants were offered a token compensation for their participation.

Participants

A total of 309 qualified participants completed the survey. All participants had direct responsibilities for software strategy, development and quality. Participants were from 5 continents providing a global perspective.

APPENDIX B — SURVEY DETAILS

This survey was performed by Dimensional Research. Here are the profiles of the respondents.

Company size

100 - 1,000 employees	22%
1,000 - 5,000 employees	33%
5,000 - 10,000 employees	16%
More than 10,000 employees	29%

Titles

Executive (VP, GM, C-level)	17%
Director	29%
Manager	54%

Primary Industry

Technology - Software	29%
Financial services and insurance	15%
Healthcare	9%
Manufacturing	8%
Services	7%
Government	6%
Retail / E-commerce	6%
Telecommunications	6%
Technology - Other	5%
Education	5%
Other	3%

Country

United States or Canada	63%
Europe	22%
Asia	5%
Mexico, Central or South America	5%
Middle East or Africa	3%
Australia or New Zealand	2%

Number of Developers

20 - 50 devs	28%
50 - 100 devs	19%
100 - 250 devs	20%
More than 250 devs	32%