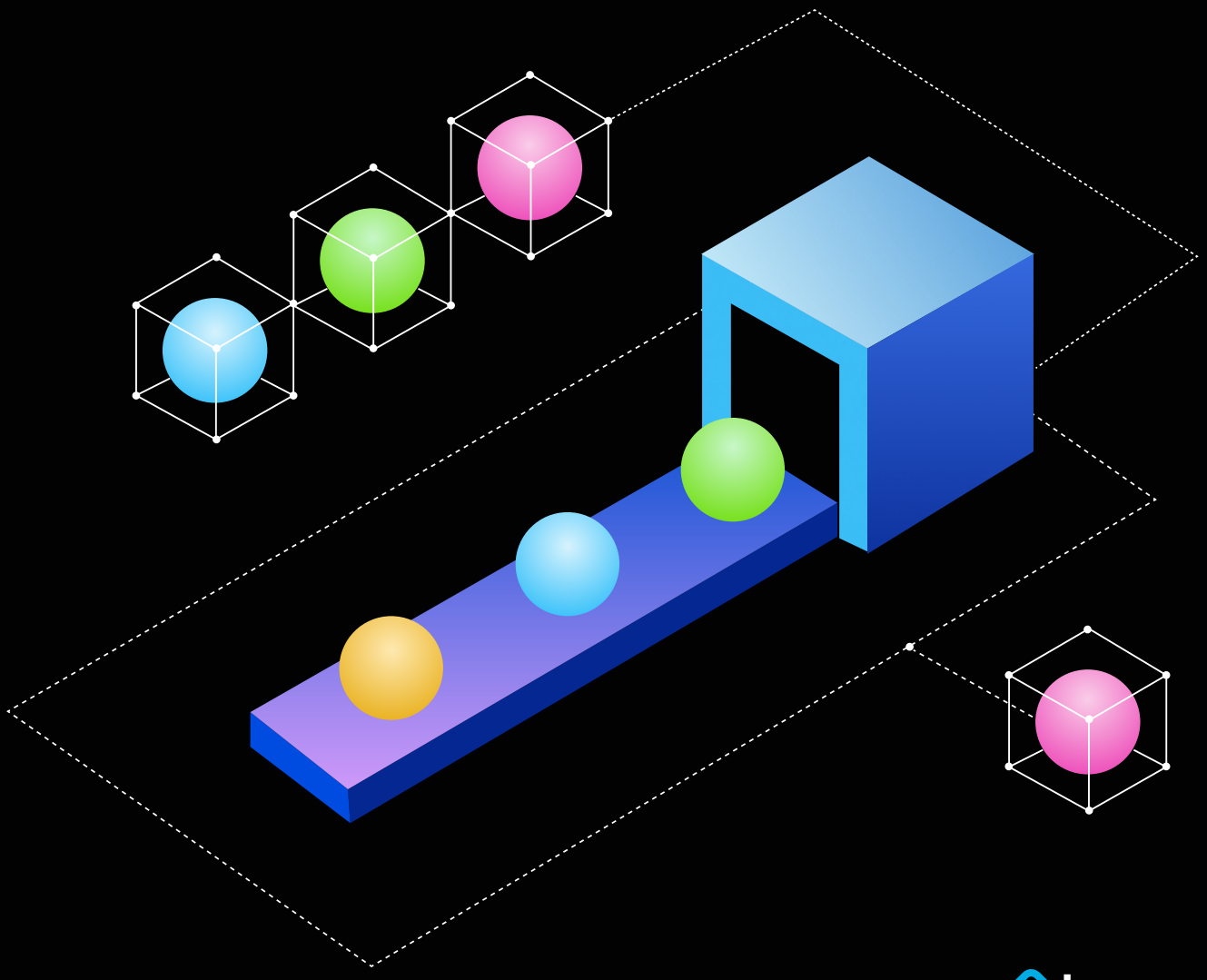
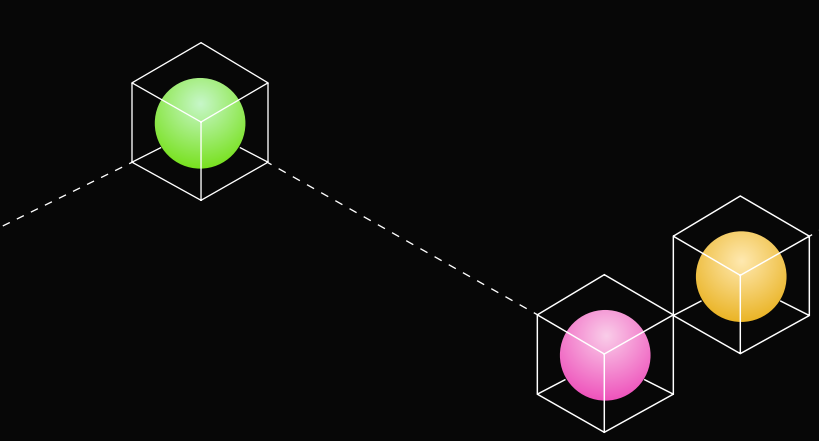
THE STATE OF

Software Delivery 2025



harness



THE STATE OF
Software Delivery
2025

02 The developer crisis is intensifying

03 Developer toil disrupts productivity

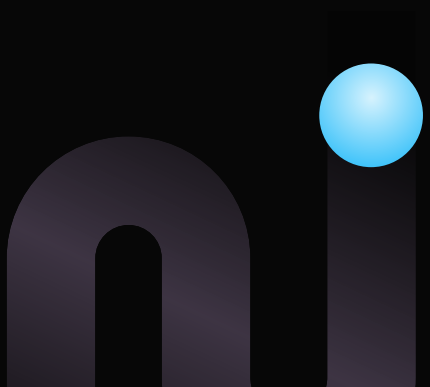
05 AI success, setbacks & surprises

09 Shadow AI is the new shadow IT

11 Unlocking value across the SDLC

14 AI is coming for your job...or not

16 Your next steps forward



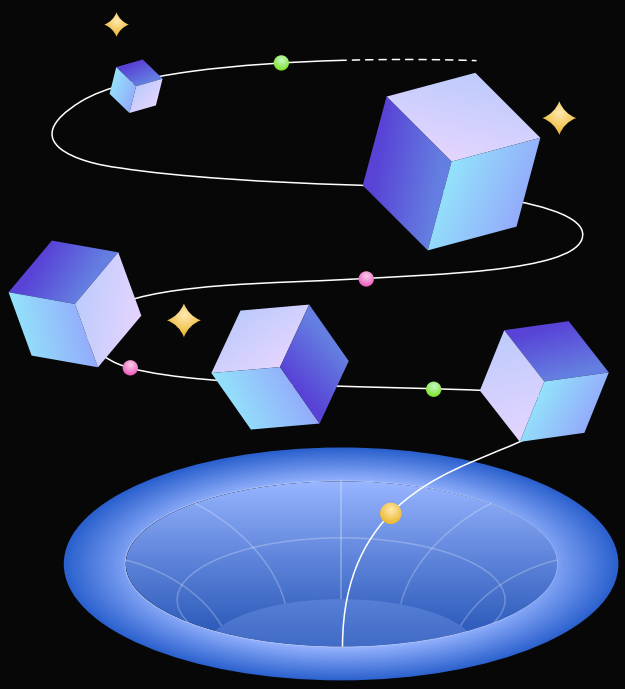
harness

The developer crisis is intensifying.

Relentless market demands have fundamentally transformed the role of software developers, oftentimes creating impossible delivery expectations. Where developers once focused primarily on writing and maintaining code, they now navigate an ever-expanding scope driven by their users' insatiable appetite for rapid innovation. In the last 24 months, AI-powered development tools have emerged as a potential solution.

We surveyed 500 engineering leaders and practitioners to explore how well organizations are adopting these tools, identify current challenges, and solidify a true path for success that incorporates AI in modern software delivery.

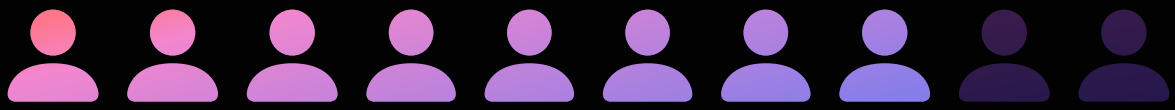
Today's developers find themselves wearing multiple hats – they're expected to be security experts implementing robust safeguards, operations specialists managing complex infrastructure, performance engineers optimizing system efficiency, and UX advocates ensuring seamless user experiences.



And while this consolidation of responsibilities can improve efficiency, it has dramatically increased the cognitive load on developers. **It also doesn't help that the majority of developers are burdened with legacy processes that result in a deluge of toil.**

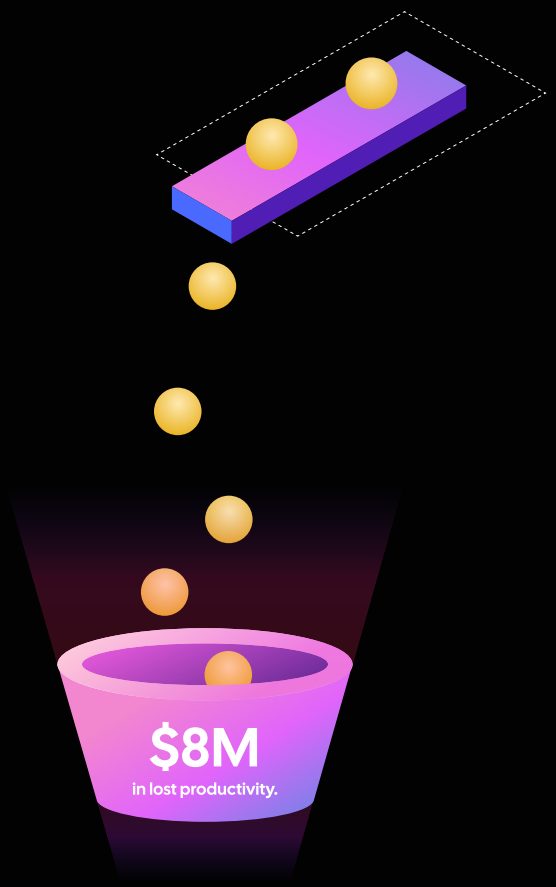
Developer toil disrupts productivity

78% of developers spend at least 30% of their time on manual, repetitive tasks.

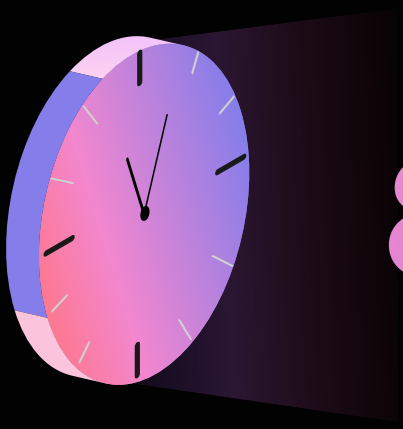


This includes things like writing compliance policies, quality assurance testing, or handling error remediation. **Organizations often exacerbate these challenges by failing to recognize the cumulative toil of constant context switching and cognitive overload.** They also fail to quantify how developer toil is hindering profitability.

For example, the average salary of the developers we surveyed is $107,599. So if 30% of their job is made up of redundant tasks, that equates to $32,280 of wasted investment in each developer. When you consider we surveyed organizations with at least 250 developers, we're talking at least **$8M in lost productivity annually per engineering team we surveyed.**



Beyond the financial waste, the drive for efficiency can actually lead to burnout as most are working overtime.



88% of developers work more than 40 hours/week

And when asked about the impact working overtime has on their workplace wellbeing and personal lives, they were pretty clear that it...

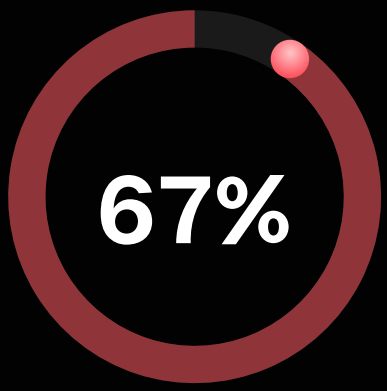
Creates an unhealthy work/life balance	50%
Increases burnout	41%
Increases stress and anxiety levels	48%
Steals time they can spend with family and friends	42%
Entices them to leave the organization	46%

AI success, setbacks & surprises

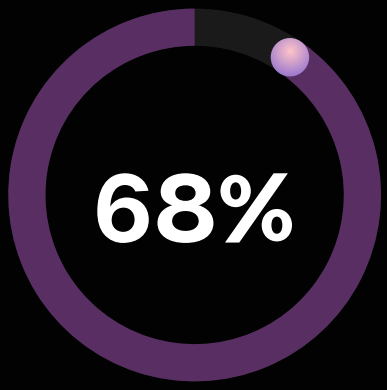
To combat this predicament, the rise of AI-powered code generation tools represent a significant shift in how developers manage their expanding responsibilities. At their core, AI CodeGen tools offer a form of intelligent augmentation that addresses the growing complexity of modern software development.

98% of developers believe AI tools are a great way to reduce burnout

However, despite this positive sentiment, adopting AI codegen is far from being a risk-free endeavor. 92% of developers stated that while AI tools increase the volume of code shipped into production, it also increases the "blast radius" from bad deployments. And that's just the tip of the iceberg.



of developers spend more time debugging AI generated code



of developers spend more time resolving security vulnerabilities

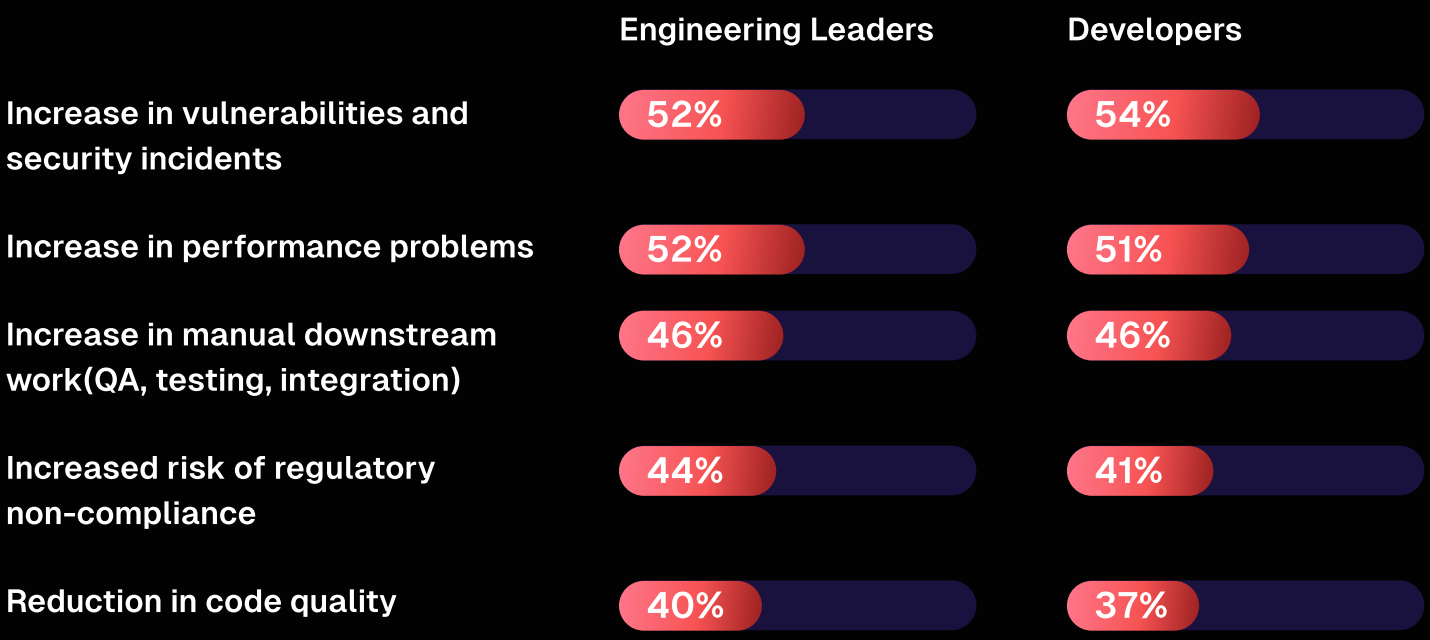
Contrary to popular belief, developers must invest significant time in understanding, debugging, and refining AI-generated code. AI systems may also generate code that includes outdated dependencies or insecure coding patterns, requiring developers to spend time updating and patching these vulnerabilities.

59%

of developers experience problems with deployments at least half of the time when using AI coding tools.

The adoption of AI codegen tools thus far has actually resulted in a shift in the developer workload–**while AI accelerates initial code production, it creates new demands around code review, security validation, and quality assurance.** This increased verification overhead arguably offsets a considerable amount of the productivity gains.

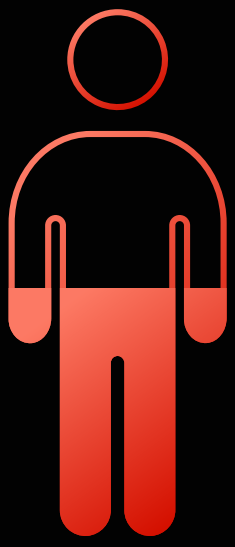
Engineering leaders also share some notable concerns about the increased use of AI code generation tools. When asked about the impact working overtime has on their workplace wellbeing and personal lives, they were pretty clear that it...



Shadow AI is the new shadow IT

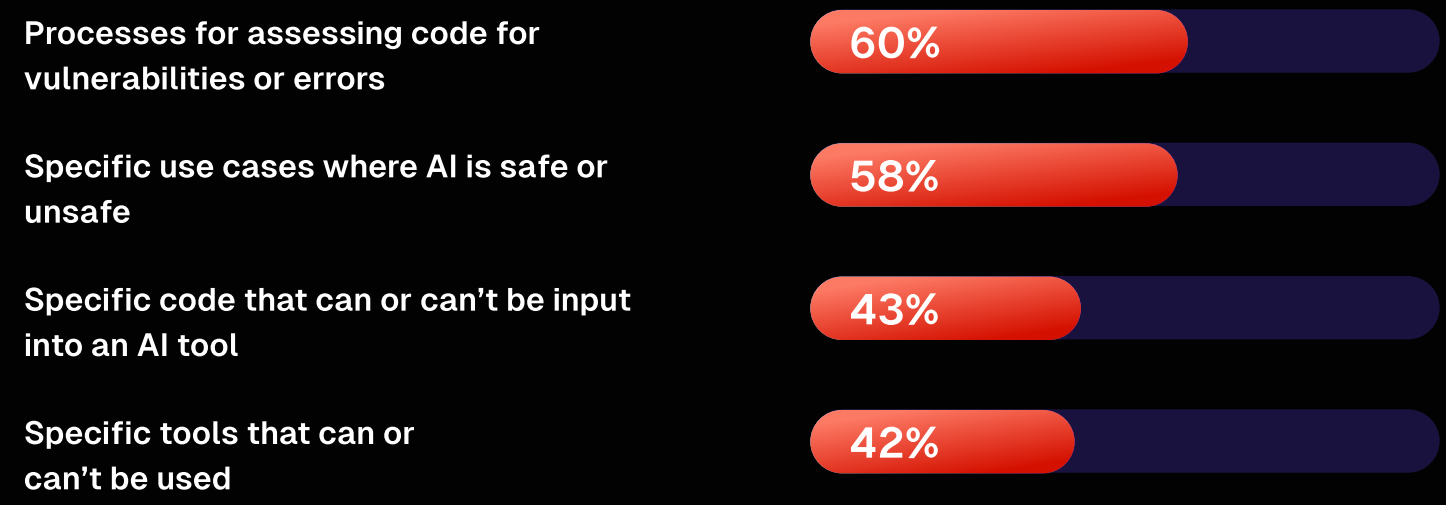
Perhaps the most alarming observation was around the use of company-approved coding tools - or lack thereof. The unauthorized adoption of AI codegen tools creates significant shadow IT challenges that extend far beyond immediate security concerns.

Shadow AI usage raises serious compliance and intellectual property concerns as sensitive code snippets or business rules might be unknowingly shared with external AI services without proper governance or legal review - ultimately they can't track the origin of AI-generated code, nor can they ensure consistent security standards across teams.



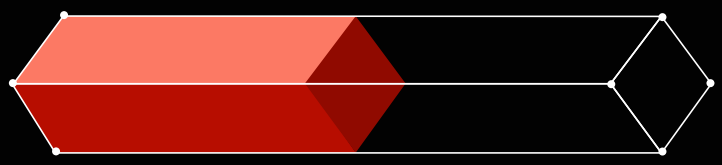
52% of developers don't use AI tools provided by IT

To combat this, company policies need to explicitly address rogue AI usage. Engineering leaders identified critical gaps regarding internal AI tooling policies:



We're just arriving at the doorstep of the AI era in software delivery. Though initially promising, there are some hurdles that hinder the full potential of AI solutions. While **81% of engineering leaders and 83% of developers** believe that software development has become more efficient since introducing AI tools, few are actually measuring the impact - good or bad.

60%

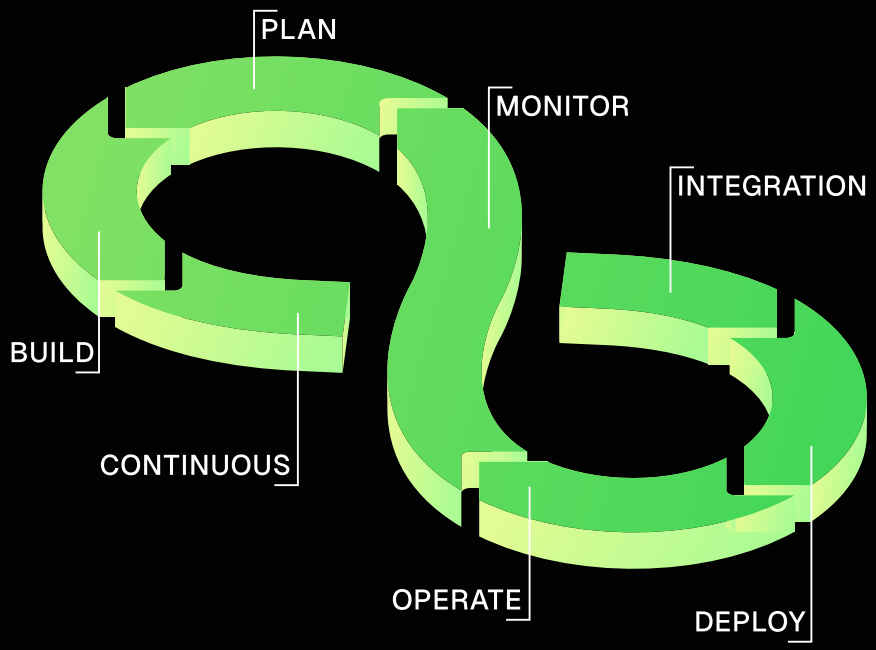


of respondents do not currently evaluate the effectiveness of their AI coding tools

Unlocking value across the SDLC

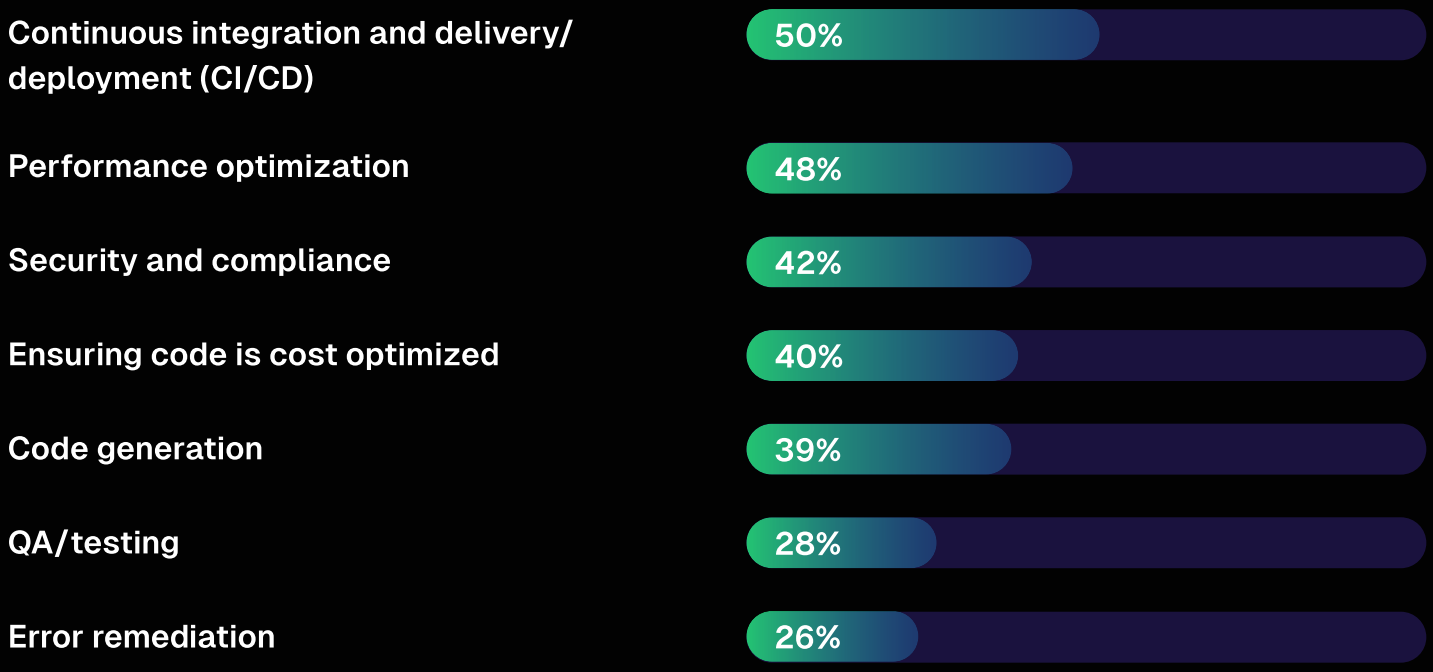
While code generation has garnered the lionshare of attention, it represents just one facet of how AI is transforming the software delivery lifecycle. The true potential lies in its ability to assist developers across the breadth of their responsibilities – from code to production, and everything in between.

95% of engineering leaders and 96% of developers say the full benefits of AI assisted software development will never be realized until their use extends to the entire software delivery lifecycle.

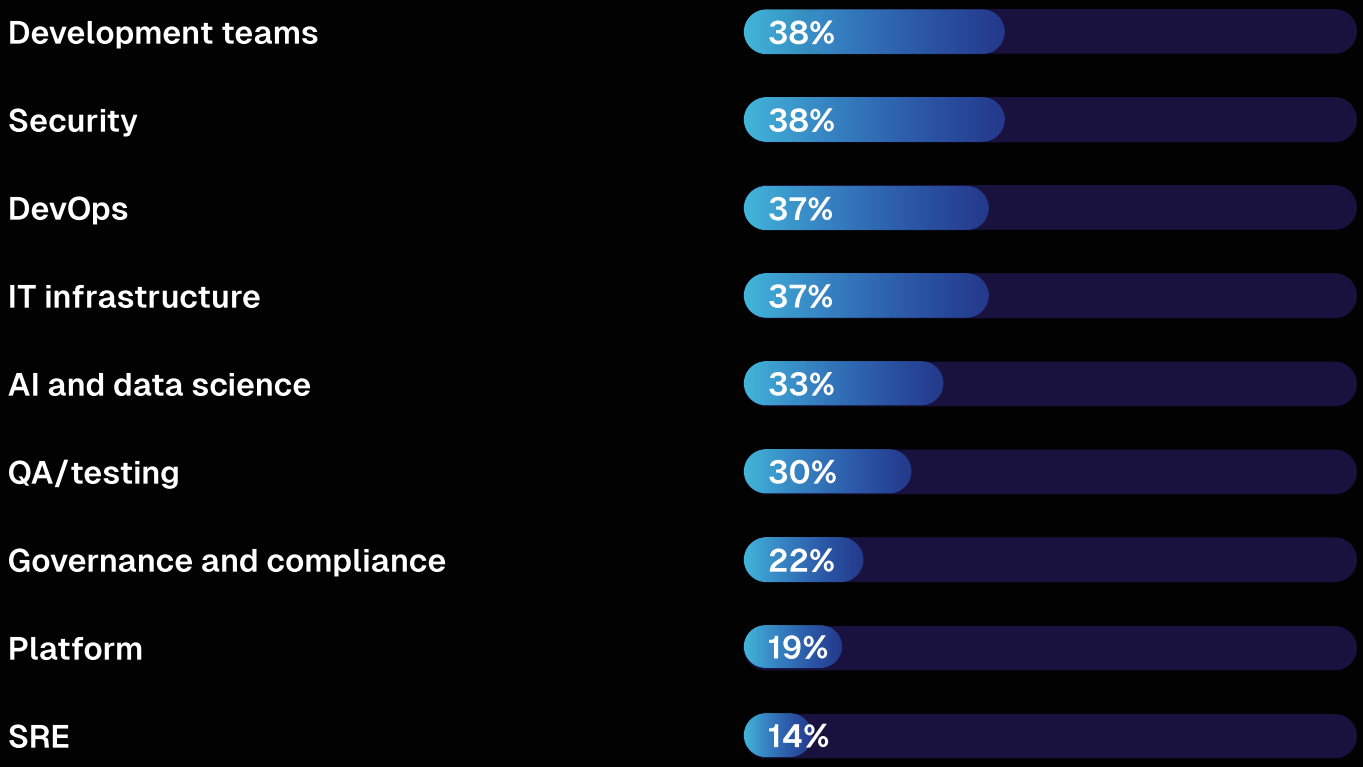


Supporting developers across the entire SDLC not only improves productivity but also significantly reduces their cognitive burden, leading to more sustainable and effective methods. But where exactly are organizations looking to make their next investment?

Engineering leaders identified where their plan to implement AI over the next 12 months:



They also identified which teams they expect to make the largest share of investment:

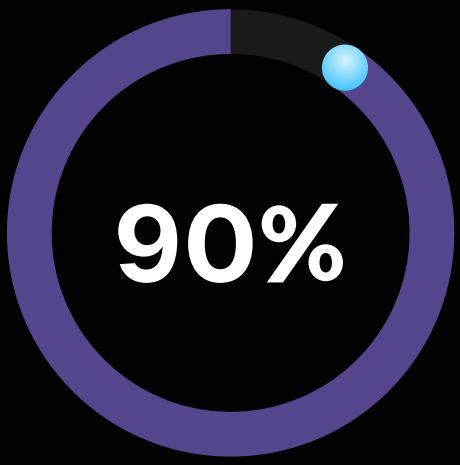


At first glance there isn't a clear priority in terms of department level funding. The reality is that today's engineering leaders have a narrow understanding of AI's potential in software development, focusing primarily on code generation while underestimating the need for complimentary AI capabilities across the development lifecycle.

This knowledge gap creates a strategic blindspot where organizations may invest heavily in code generation tools while missing opportunities to leverage AI for more substantial improvements in development efficiency, system reliability, test strategy optimization, and production system monitoring.

AI is coming for your job... or not

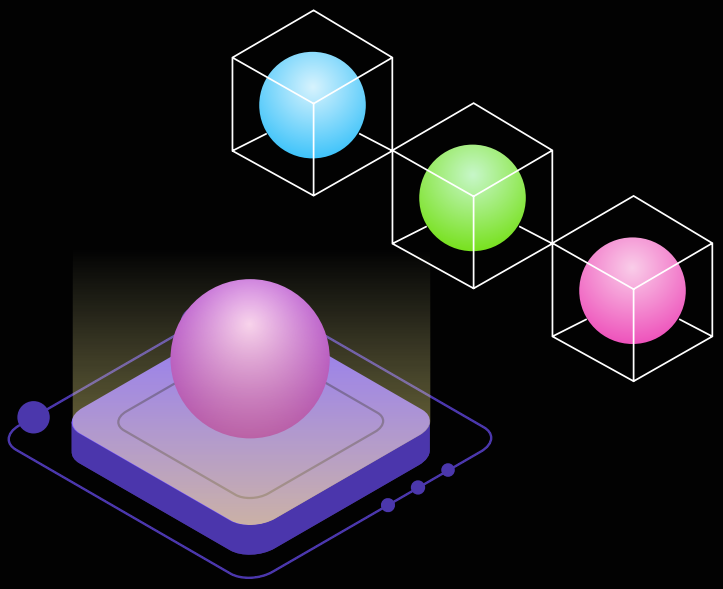
Last but not least, we still haven't addressed the elephant in the room in terms of the net impact to developer staffing. Regardless of AI's real or perceived value, the threat of job displacement is materializing and has become top of mind throughout the engineering world.

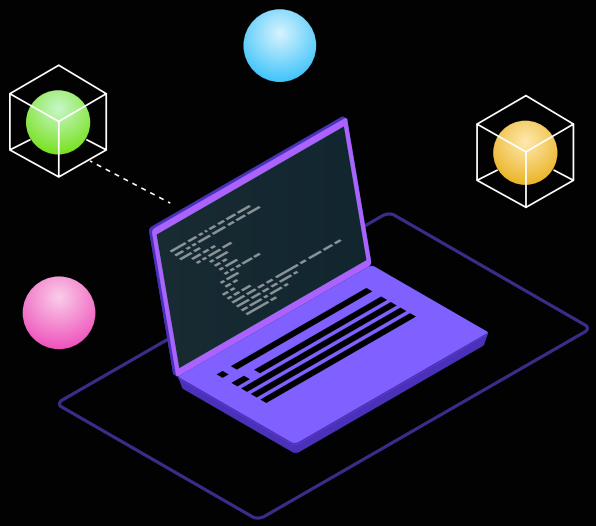


of respondents are concerned that AI tools will replace developers

The truth is, the role of software developers extends far beyond writing code. It includes designing system architecture, problem-solving, and business logic interpretation.

At the very least developers need contextual knowledge and creativity. And one thing AI tools currently lack is the ability to grasp topics in a broader business context.





As AI tools become more prevalent, the role of developers will evolve to include new functions.

AI prompt engineering, output validation, and capabilities integration into development workflows are just a few examples.

So rather than displacement, we're seeing a transformation of their role where AI handles routine tasks and frees developers to focus on higher-value activities.

Your next steps forward

Before we get too ahead of ourselves, let's recap our findings and highlight a few key takeaways. That way you have a clear summary of AI's impact within software delivery. Ultimately, the integration of AI into development workflows requires careful consideration. While AI tools can significantly reduce cognitive load, they also introduce new complexities around tool selection, integration, and governance.

And the reality is, most organizations don't have a real picture of these nuances prior to starting their journey. Companies must carefully balance the benefits of AI automation against the need for human oversight and understanding. From a desired outcomes perspective, the goal should be to use AI to augment developer capabilities rather than replace human judgment.