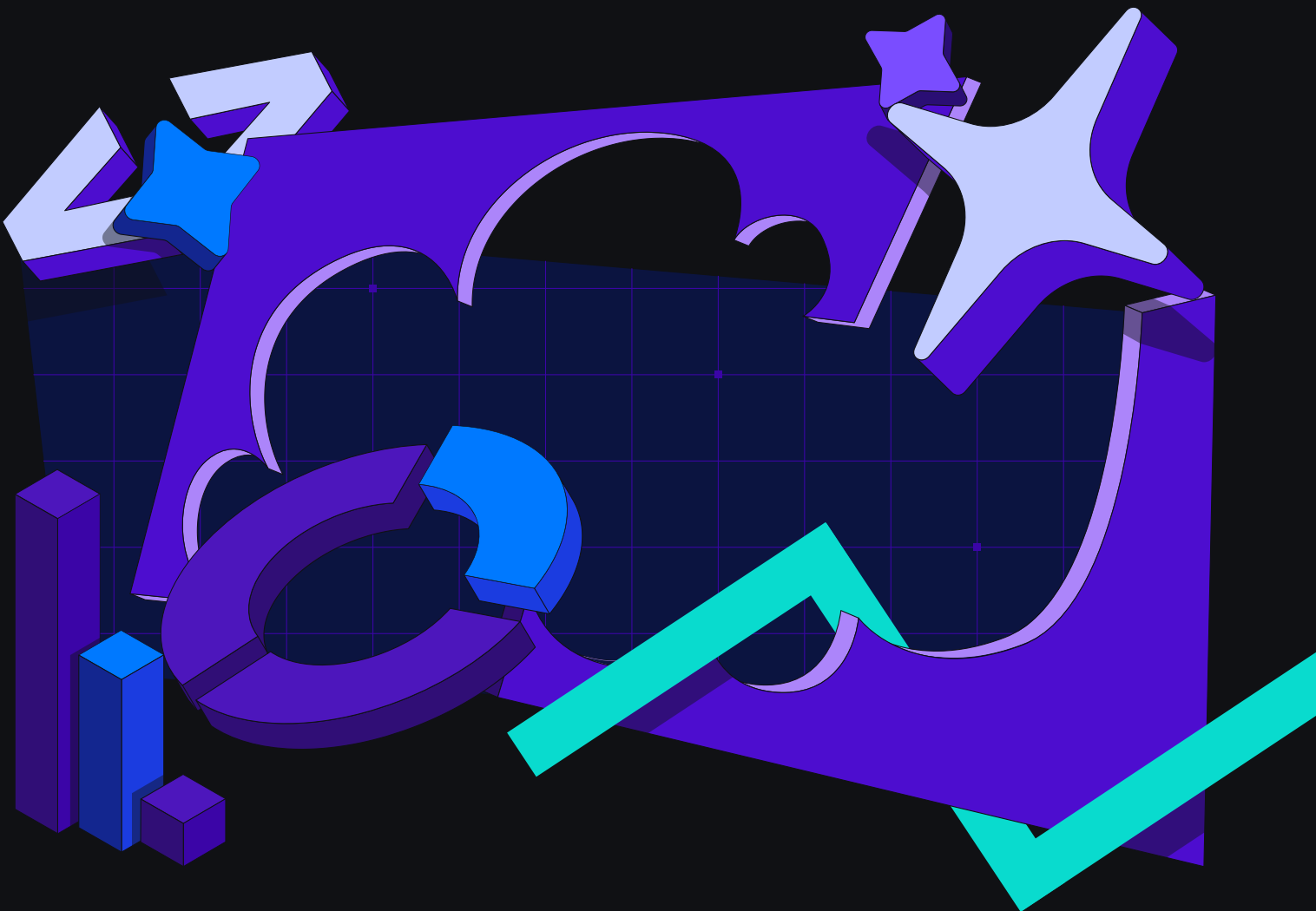


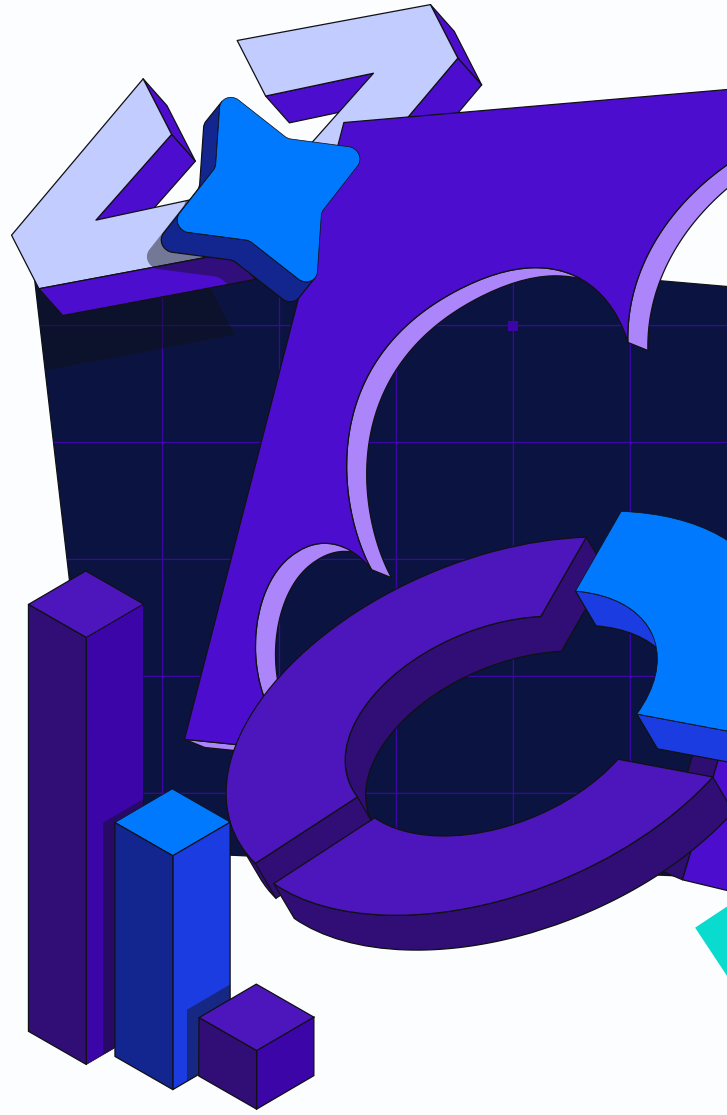
State of AI in Cloud Development 2026

The intersection of code & cloud in AI-native application development



Contents

Introduction	3
Audience demographics	4
Research methodology	4
Main takeaways	5
AI now plays a role at nearly every stage of development	6
AI agents are becoming increasingly autonomous	7
AI writes more code, but nearly 40% never ships	8
Teams that ship most of their AI code see bigger output gains	9
Developers are concerned about giving AI access to the cloud. They do it anyway.	10
The cloud is the new bottleneck	12
Developers and managers don't see eye to eye on AI	14
One AI coding tool isn't enough	15
Conclusion	16
Recommendations	17
About Dimensional Research	18
About LocalStack	18



Introduction

AI adoption is raising new questions about how enterprises are using the tech stack to build cloud applications.

Traditionally, enterprise software teams have approached public cloud infrastructure cautiously, to prevent security risks and keep cloud costs under control.

Are those same concerns slowing down the adoption of AI? Are the promises of AI velocity causing software developers to think differently about the cloud?

Most importantly, how has the development experience adapted to the changing needs of an increasingly AI-native software development lifecycle (SDLC)?

To answer these and other questions about the impact of AI, Dimensional Research partnered with LocalStack to survey those responsible for building cloud applications at large enterprises.

This report highlights the findings from that survey.

Audience demographics

A total of 309 qualified participants completed the survey.

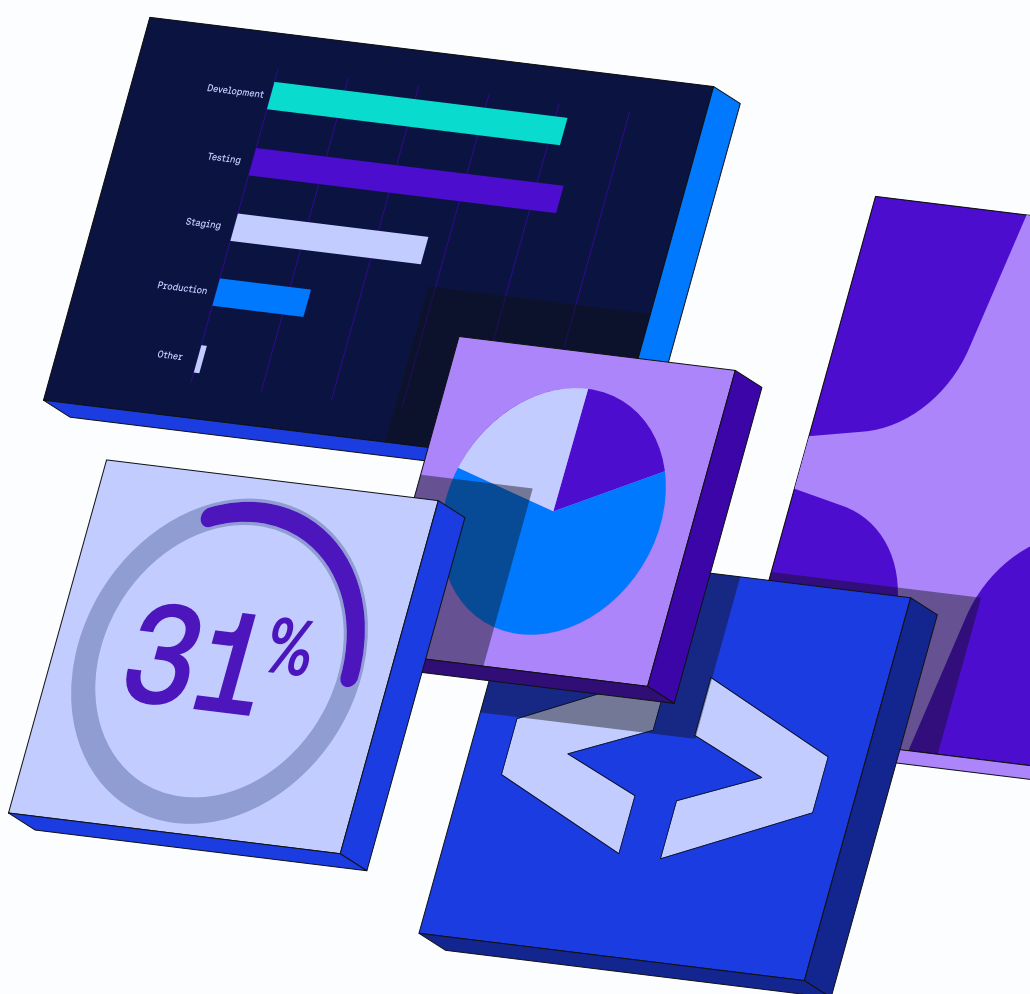
At the time of the survey, all participants were responsible for developing applications that would be deployed to public cloud infrastructure. All participants lived in the United States or Canada and worked in a hands-on development or team manager role at a company with more than 1,000 employees.

Research methodology

The primary research goal was to capture data on how AI is changing cloud application development, including trends, attitudes and experiences.

Independent sources of development stakeholders were invited to participate in an online survey. Questions were asked on a variety of topics related to AI coding tools, Agentic AI, and cloud development environments. The survey was fielded in English.

Responses were captured September 7–14, 2026.



Main takeaways

The push to implement AI across the SDLC continues despite reservations about the quality of output, the capabilities of the development tech stack, and the security implications of AI agents building in the cloud.

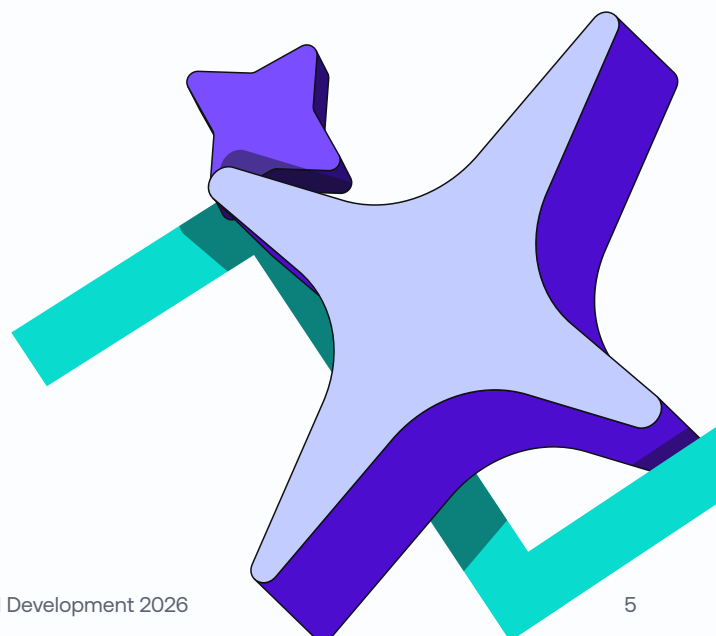
Key findings include:

65% of teams now generate at least twice as much code as they did before adopting AI, and nearly one-third generate five times as much or more.

Nearly 40% of AI-generated code never reaches production, based on respondents' estimates. More respondents estimated their teams ship less than half of AI-generated code to production than those who ship 80% or more.

Nearly 80% of respondents said they give AI agents access to cloud development environments, even though 97% cited serious concerns about doing so.

87% wait 2 minutes or more for cloud staging environments to reflect each code update, and only 9% say their staging environment fully matches production.



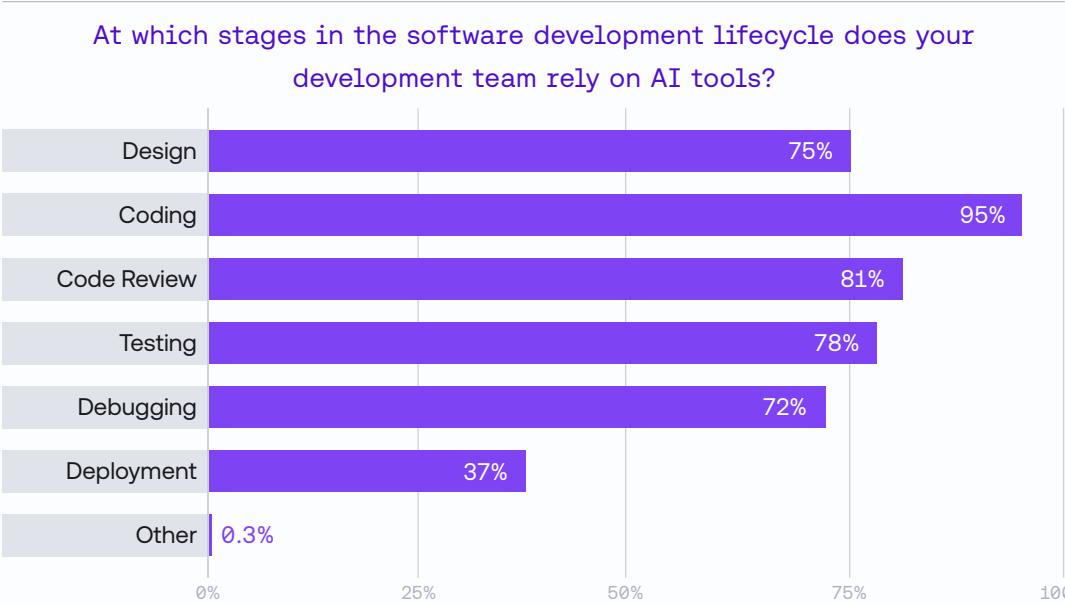
AI now plays a role at nearly every stage of development

Software teams are clearly seeing the benefits of AI for speeding up the creation of applications.

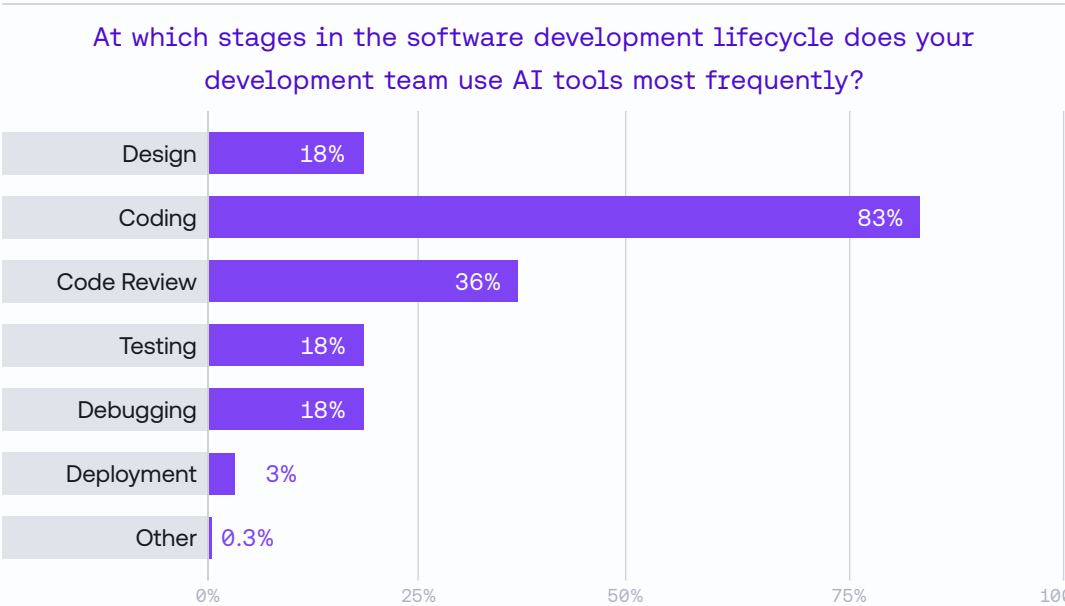
65% of respondents claim they generate at least twice as much code since adopting AI, and 32% now create at least five times as much code.

For the majority of respondents, AI plays at least some role at every stage of software development aside from deployment.

However, it's clear that AI is most trusted for its coding capabilities. When asked how they use AI most frequently, 83% said coding. No other activity was cited by more than 36% of respondents.



32%
say they create at least 5x more code since adopting AI



AI agents are becoming increasingly autonomous

Software teams are leaning into autonomous agentic AI. Nearly half (49%) of respondents said their organization has adopted fully autonomous agents that do not rely on human prompts to take action. Considering that only slightly more respondents (52%) said their organizations use agents that rely on human prompts, these results indicate that autonomy is becoming increasingly common in enterprise agentic software development.

However, the transition to fully autonomous agentic AI remains a work in progress. While 41% said AI agents occasionally run autonomously without waiting on human prompts, only 8% of respondents said AI agents do so frequently.

41%

AI agents occasionally run autonomously without waiting on human prompts

8%

AI agents frequently run autonomously without waiting on human prompts

Those at larger organizations are more likely to run autonomous AI agents. 54% of respondents at organizations with more than 5,000 employees reported running autonomous AI agents that run without relying on human prompts, while just 40% of those at smaller organizations said the same.

Teams that run autonomous agents also generate more code. 40% of respondents whose AI agents can run without waiting on human prompts say their team generates at least five times as much code as before adopting AI, compared with 26% of those without autonomous agents. They are also more confident pushing AI-generated code to production (26% vs. 18%).

AI writes more code, but nearly 40% never ships

As AI increases the volume of code, development teams struggle with its quality. Nearly half of respondents (47%) agreed that the quality of AI-generated code is significantly worse than that of human-generated code.

Only 22% of respondents are more confident pushing AI-generated code to production, compared with 61% who are more confident pushing human-generated code.

More than half (54%) said they encounter more bugs in AI-generated code, while 31% said they see fewer bugs in AI-generated code.

On debugging, 47% of respondents said human-generated code is easier, compared with 30% who say AI-generated code is easier.

Quality concerns may help explain why so much AI-generated code never reaches production. When asked to estimate what percentage of AI-generated code their teams release to production, responses varied widely.

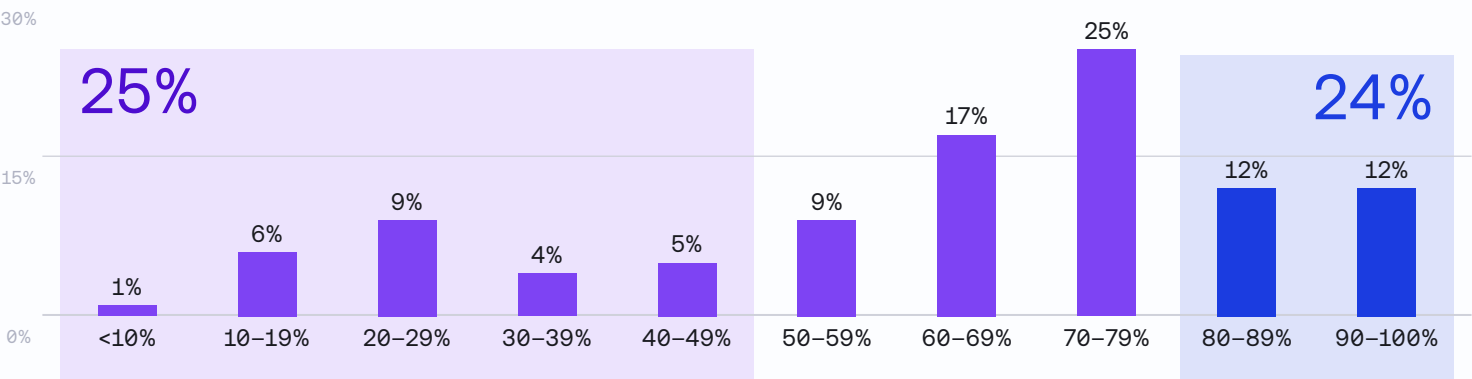
Based on respondents' estimates, only 61% of AI-generated code is released into production, meaning nearly 40% never ships.

More respondents (25%) estimated that they release less than half of their AI-generated code to production than those who said they release 80% or more (24% of respondents).

25%
ship less than half of AI-generated code to production.

24%
ship 80% or more of AI-generated code to production.

To the best of your knowledge, what percentage of AI-generated code that your team produces is released to production?



Teams that ship most of their AI code see bigger output gains

The survey also indicated a correlation between maturity in AI and an ability to release code to production.

Among respondents who release 80% to 100% of their AI-generated code, 57% say their team generates at least five times as much code as before adopting AI. By comparison, only 24% of those who ship less than half of AI-generated code claim the same gains in code creation.

Meanwhile, those that frequently run autonomous agents also release more of their AI-generated code to production. Among those who release 80% to 100% of AI-generated code to production, 14% said their AI agents frequently operate without relying on human prompts, the highest of any group. Among those who said less than 50% of their AI-generated code makes it to production, only 3% frequently run autonomous AI agents.

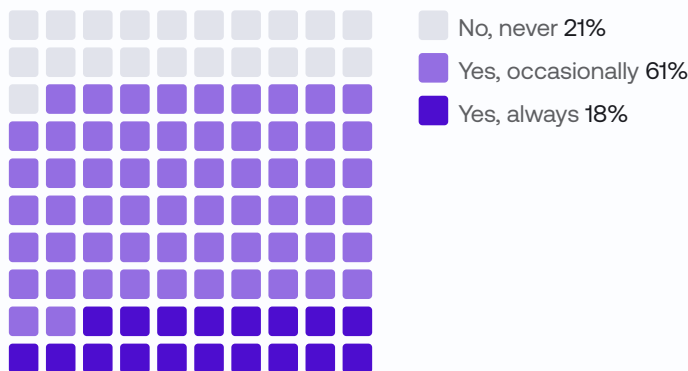
Developers are concerned about giving AI access to the cloud. They do it anyway.

Concerns about security, application reliability, and cloud costs haven't stopped software teams from providing AI agents access to the cloud.

Most respondents (79%) said they give AI agents access to cloud environments for development, testing, or staging. That includes 61% who do so occasionally, and 18% who always do.

By comparison, only 45% of respondents said all human developers have access to cloud environments.

In your development environment, do AI agents have access to a cloud environment for developing, testing, or staging application code?



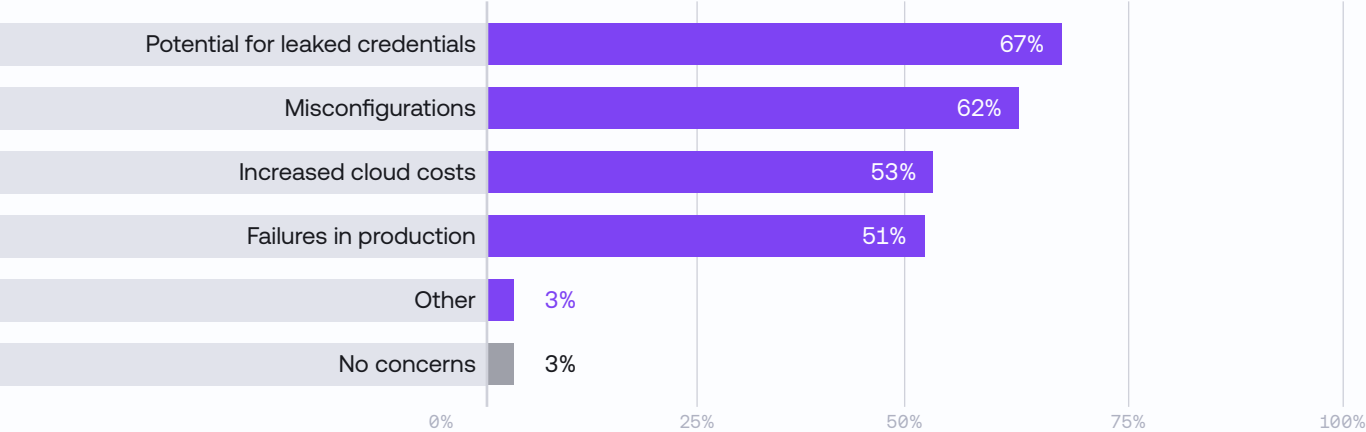
79%
give AI agents access to cloud environments

Those at larger organizations are more likely to grant AI agents access to the cloud: 84% of those at companies with more than 5,000 employees do so, compared with 72% at organizations with 1,000–5,000 employees.

Despite this trend, 97% of respondents reported concerns about giving cloud access to AI agents. Security risks (leaked credentials and misconfigurations) were top of mind, followed by cloud costs and failures in production.

97%
reported concerns about giving cloud access to AI agents

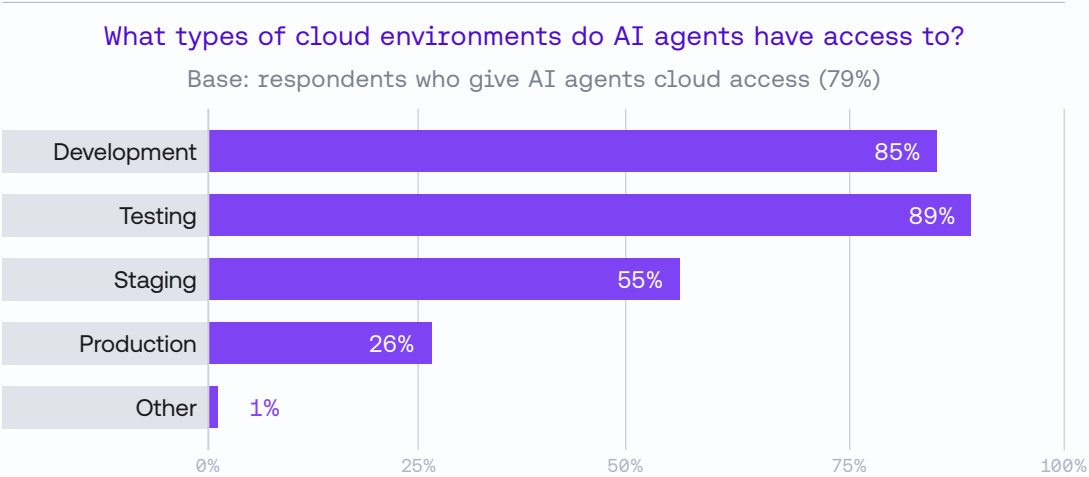
What concerns do you have about providing AI agents with access to cloud environments?



Even 92% of those who said they always grant AI access to the cloud expressed concerns about doing so:



For a significant number of respondents, AI agents have access to cloud environments from development through production. While agents were most likely to have access to testing environments, as cited by 89% of those who grant access, 26% said AI has access to production.



The cloud is the new bottleneck

While most software teams give AI agents access to cloud development environments, many report frustrations with the speed and reliability of those environments.

Building, iterating, and debugging cloud applications often requires testing frequent code updates in a staging environment. While the amount of time varies by application, each code update takes time to deploy due to the latency and complexity of provisioning remote infrastructure.

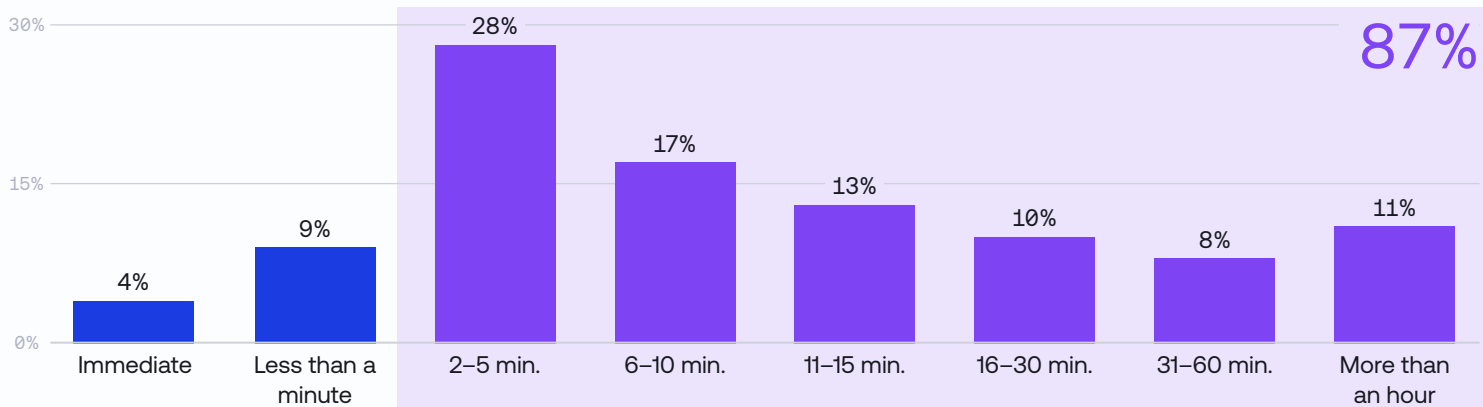
The survey found that these delays can be significant. When asked how long it takes for a cloud staging environment to provision after pushing a code update:

- 87% wait 2 minutes or more
- 42% wait longer than 10 minutes
- 29% wait longer than 15 minutes
- 19% wait longer than 30 minutes
- 11% wait longer than an hour

42%

wait longer than 10 minutes to validate each code update

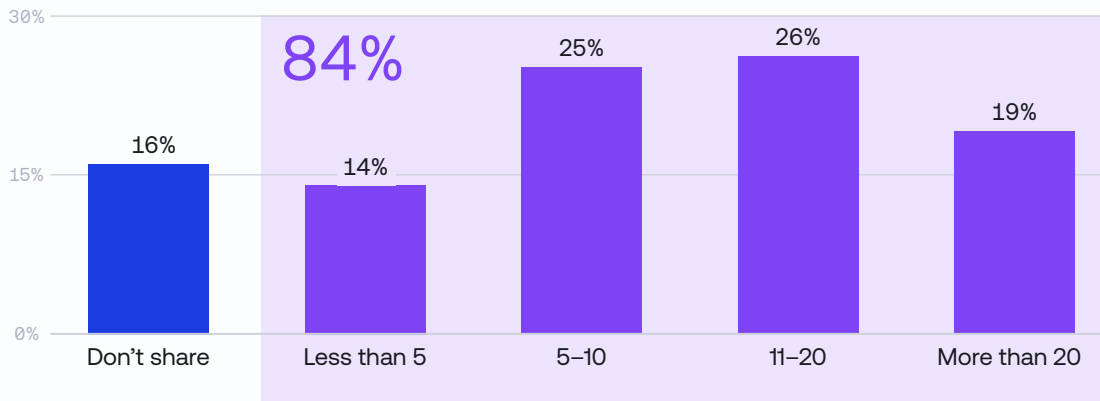
When pushing code updates, how long does it typically take for a cloud staging environment to provision to reflect the changes?



Developers also reported sharing access to cloud staging environments for developing and testing applications. Shared environments are a common way to balance the development team's needs against the overhead and risk of managing permissions, configurations, and costs for non-production environments.

In fact, 84% of respondents said they rely on shared staging environments for their cloud development work. Most (70%) share environments with 5 or more developers, while 45% share with more than 10.

How many developers in your organization typically share a single staging cloud environment?



45%

share staging environments with more than 10 developers

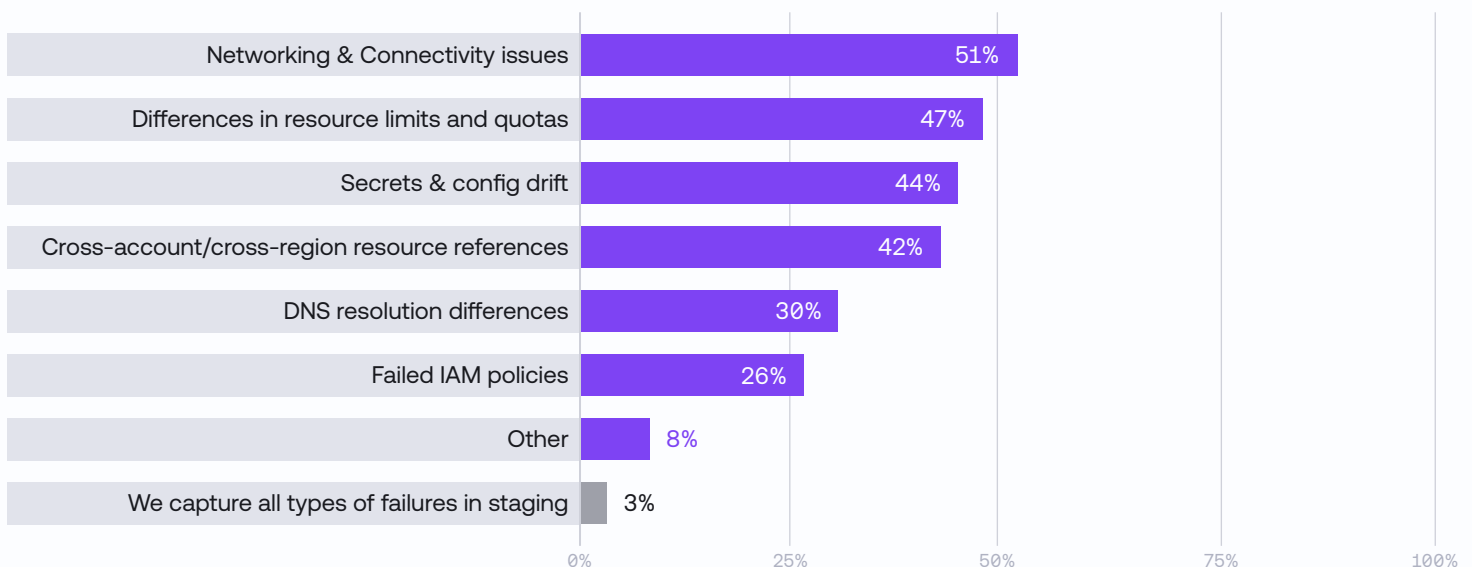
Developers often take turns working in shared staging environments to prevent pushing conflicting code updates that can crash the environment.

Even so, 82% of those who share staging environments said downtime disrupts their work at least once per week. For 27% of respondents, these disruptions occur 3 or more times per week.

Staging environments notoriously struggle to replicate production conditions, resulting in inaccurate tests that fail to catch defects before they're released to production. Only 9% of respondents said their staging environment perfectly matches production, and 42% said it misses issues that appear in production or isn't close to replicating production at all.

Nearly all respondents (97%) report encountering failures in production that staging environments struggle to catch, ranging from networking issues to failed IAM policies:

What are the most common failures that happen in production that staging can't reliably capture?



Developers and managers don't see eye to eye on AI

Survey responses show that developers do not always share managers' views on the quality and maturity of AI in the SDLC.

Some notable examples include:

23% of developers said AI-generated code has significantly more bugs than human-generated code, while just 6% of team managers said the same.

51% of developers said human-generated code is easier to debug than AI-generated code, compared to 39% of team managers who said the same. In fact, nearly as many managers (38%) said the opposite: AI-generated code is easier to debug.

27% of developers said they feel significantly more confident shipping human-generated code to production than code created by AI, while just 15% of team managers said the same.

Managers are split on whether AI-generated code is harder to debug. Most developers say it is.

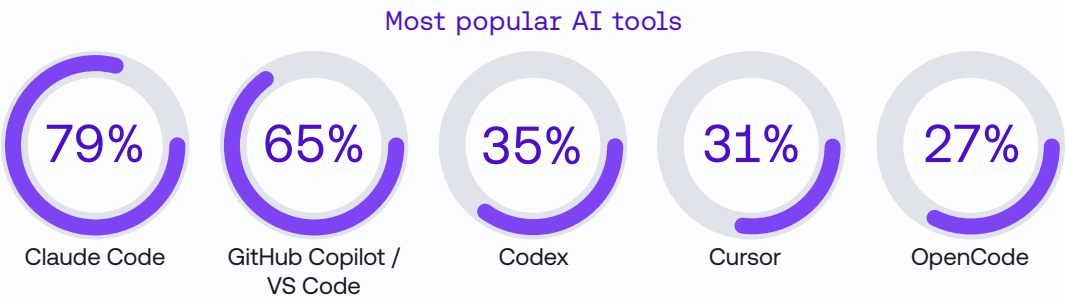
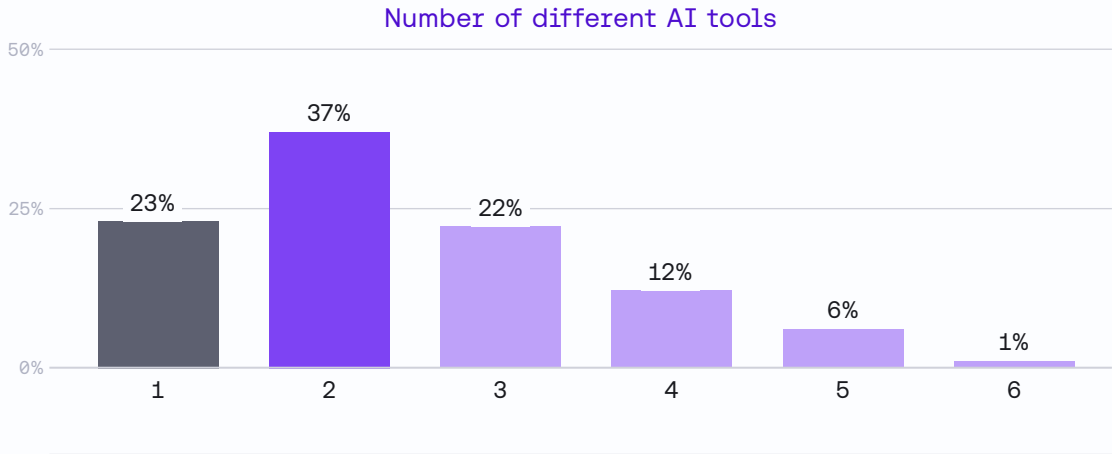
Managers were also more likely to say their organization deploys autonomous agents for software development. When asked how autonomous their AI agents are, 55% of managers said their teams run agents that operate without waiting on human prompts, compared to 44% of developers who said the same. However, developers were more likely to say their agents frequently run without waiting on human prompts (9% vs. 5%).

Despite these differences in opinion, most respondents agreed that they are not equipped to meet the expectations of leadership when it comes to cost optimization. 58% of all respondents agreed that their management teams want them to manage cloud and token costs, but they don't have the information required to do so.

One AI coding tool isn't enough

The survey also revealed several trends in how enterprise software development teams use AI.

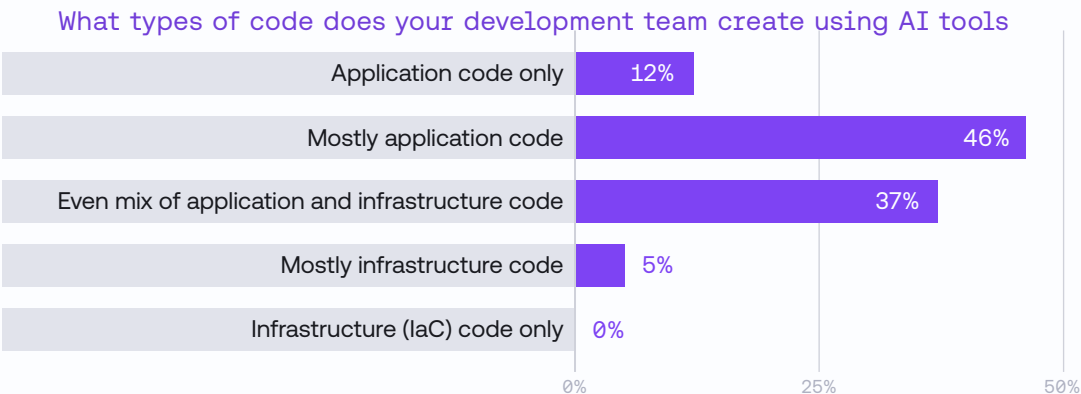
77% of respondents said their teams use 2 or more AI coding tools in support of software development, and 41% use 3 or more.



Claude Code is the most popular AI coding tool, cited by 79% of respondents. GitHub Copilot/VS Code is No. 2, at 65%. Codex (35%), Cursor (31%), and OpenCode (27%) rounded out the list.

Writing application code remains the most common use case for AI, although 42% of respondents use AI for at least as much Infrastructure as Code (IaC) as application code.

This focus on writing code aligns with how respondents characterize the role AI agents play in day-to-day operations. 41% of respondents said AI agents write code during the development phase regularly, while just 22% say AI agents regularly test and debug in the cloud staging environment.



Conclusion

The survey findings suggest that enterprise software teams are feeling pressure to implement AI across the SDLC. After years of policing developers' access, permissions, and configurations, software teams appear to have shifted their priorities to favor agentic workflows over secure cloud operations. This is happening despite the very concerns that led to restrictive cloud policies for developers in the first place.

The pressure to adopt AI may also contribute to the differences in perspective between team managers and software developers. Responses indicate that managers tend to feel more confident both in the quality of code created by AI and in the maturity of their agentic AI use. Perhaps the impression given to team managers is different from the reality of the developers' day-to-day experience using AI.

Based on respondent sentiment, concerns about the quality of AI-generated code are justified by the inability to ship that code to production. Given that respondents estimated, on average, that 61% of AI-generated code is released to production, even 2x gains in the creation of application code fall short of the potential value of agentic AI. As enterprises aim to show return on investment in AI, this waste will receive more scrutiny.

The findings on cloud development environments may provide an explanation for this shortfall. Developers have long struggled with provisioning delays, instability, and unreliable tests while building in cloud development environments. As agentic AI increases the volume of code they are responsible for validating, the cloud development environment may be standing in the way of shipping it.

The teams that fix how they validate code, not just how they write it, will be the ones that turn AI's speed into software that ships.

Recommendations

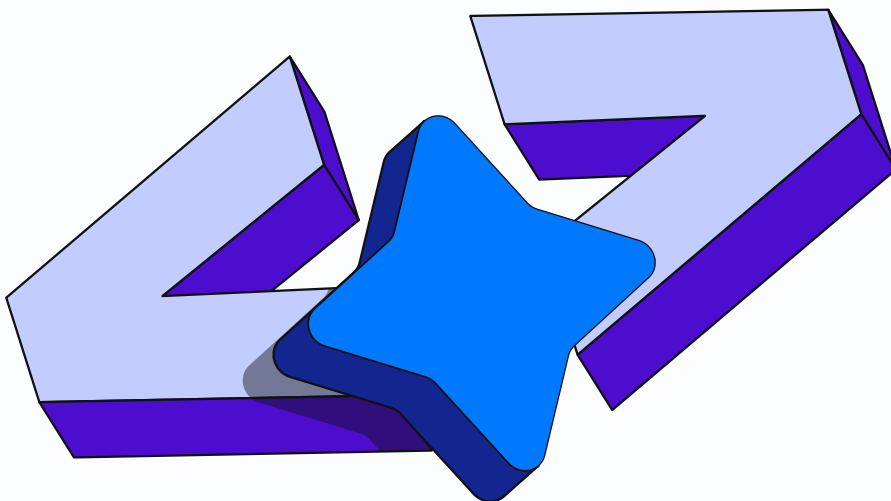
What engineering leaders can do:

Measure what ships, not what's generated. Respondents estimate that only 61% of AI-generated code reaches production. Track release rates, not just output.

Give AI agents a safe place to test. 79% of respondents already give agents cloud access, and 26% of those include production. Isolated, disposable environments let agents validate code without putting shared or production infrastructure at risk.

Shorten the validation loop. 87% wait 2 minutes or more for staging to reflect each code update. Faster feedback keeps AI's speed gains from disappearing in the queue.

Ask the people writing the code. Developers are nearly four times as likely as managers to say AI-generated code has significantly more bugs (23% vs. 6%). AI strategy should reflect both views.



About Dimensional Research

Dimensional Research® provides actionable market research for innovative 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 technology and business models used by modern businesses and their customers. For more information, visit www.dimensionalresearch.com.

About LocalStack

LocalStack is the local cloud development platform for developers and AI agents. With more than 1,500 customers and hundreds of millions of Docker pulls, LocalStack has become the de facto standard for local cloud development worldwide. LocalStack is remote-first and backed by Notable Capital. For more information, please visit www.localstack.cloud.