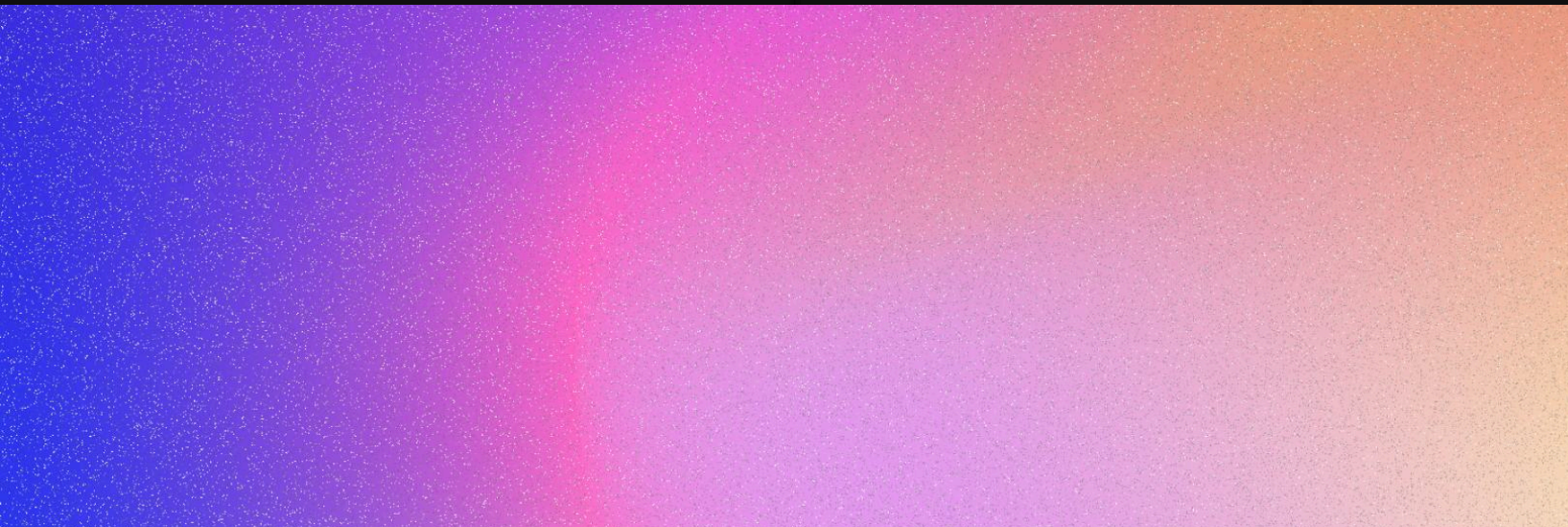
STACKLOK

State of AI Code Generation Survey

SUMMER 2025



Contents

Introduction

The following report on AI code generation pains and priorities is based on responses from more than 300 engineering leaders. Each of these leaders worked at a company with 100+ employees and led a team of 10+ developers. The intent was to better understand a cohort with accountability for scaling the use of AI tools and practices.

The intention of this report is to be as direct and objective as possible; any editorial we've offered is specifically tagged as such.

Given the pace of change in AI code generation, we intend to run more surveys like this, and we welcome your ideas for additional topics to explore. We encourage you to jump into our Discord to engage with us directly.

Section 1: Respondents

The respondents for this study were the leaders accountable for scaling AI code generation tools and practices. Every respondent managed a team of at least 10 developers in a company with 100 or more employees.

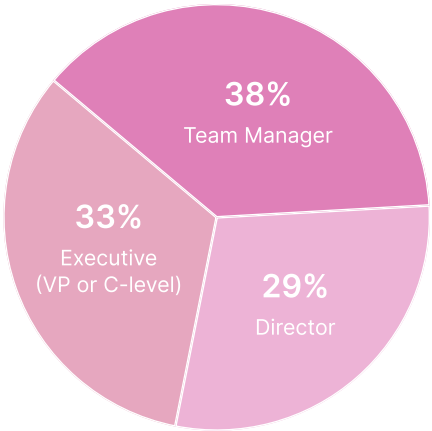


Figure 1: Job Levels

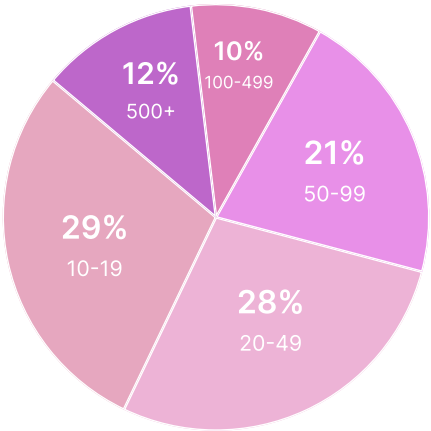


Figure 2: Number of Developers Managed

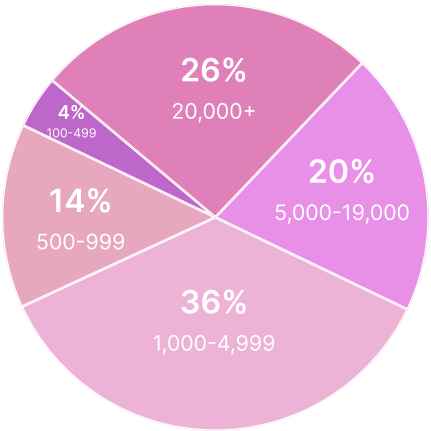


Figure 3: Company Size (Total Employees)

Section 2: Tool Use

We asked these leaders which general-purpose tools their teams were currently using to support code generation. Unsurprisingly, ChatGPT was way out in front at 83%. Gemini was second at 65%, and then there was a steeper drop-off to Claude in third at 35%.

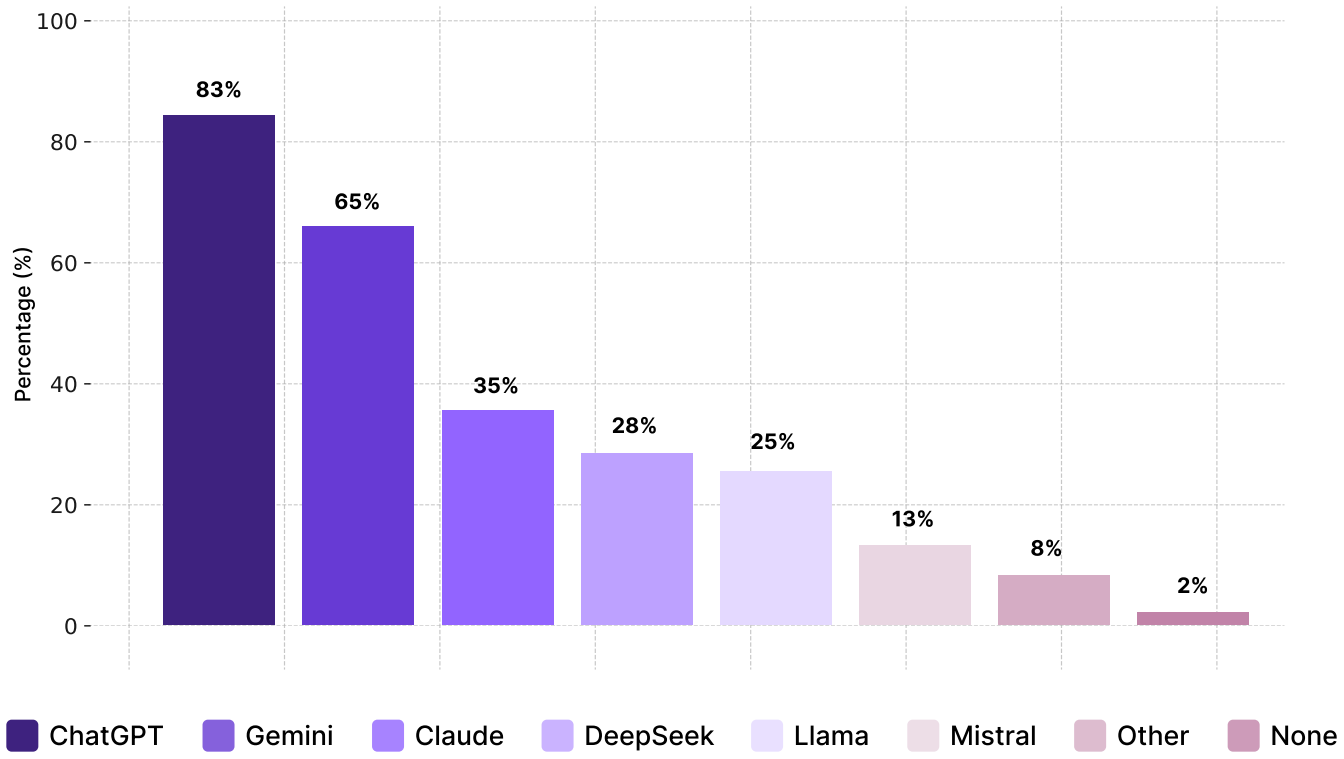


Figure 4: General-purpose code generation tool usage

Next, we asked about the use of purpose-built code generation tools. Again, there were no surprises at the top, where GitHub Copilot was well out in front while Amazon Q, Claude Code and Cursor lead the group giving chase.

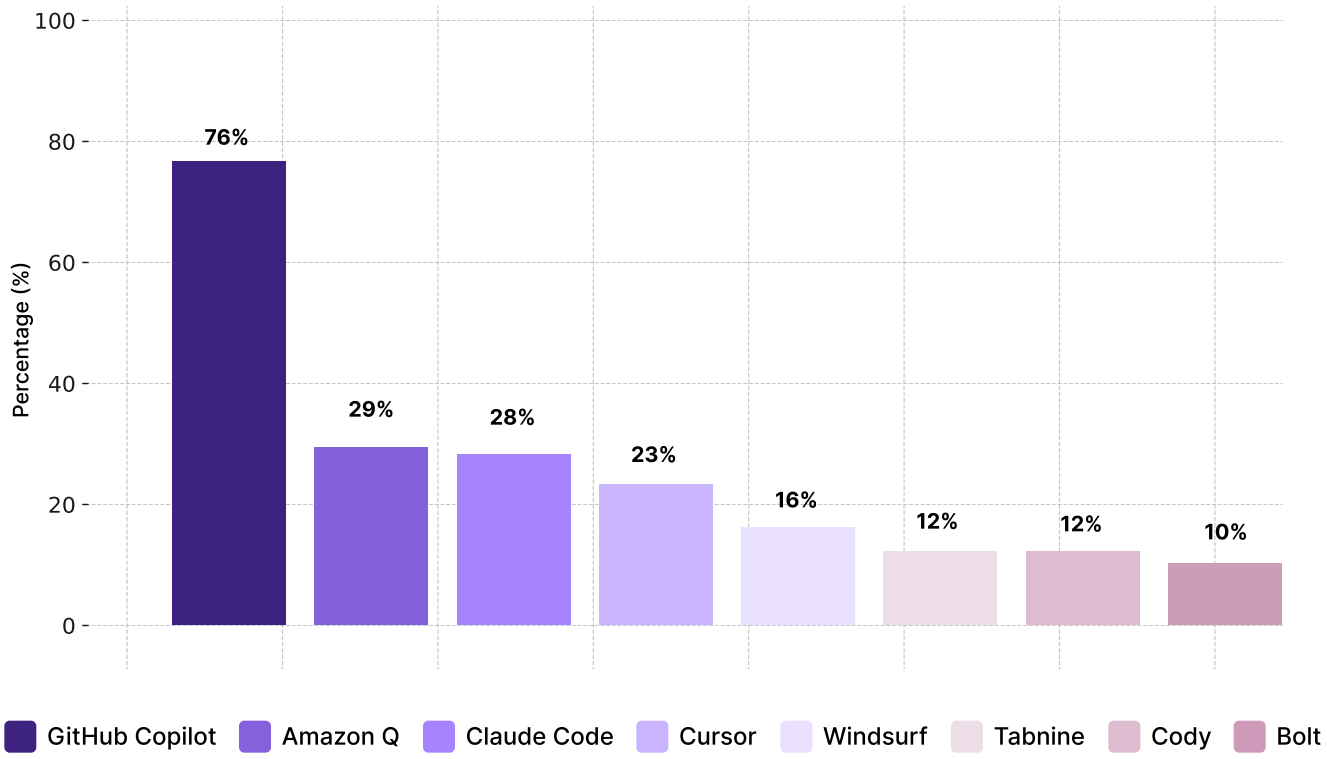


Figure 5: Purpose-built code generation tool usage

There's a difference between what tools are officially supported and those that are unofficially used (presumably because of superior benefits). The following table is dense, but it tells an interesting story: some tools see more usage despite less organization support.

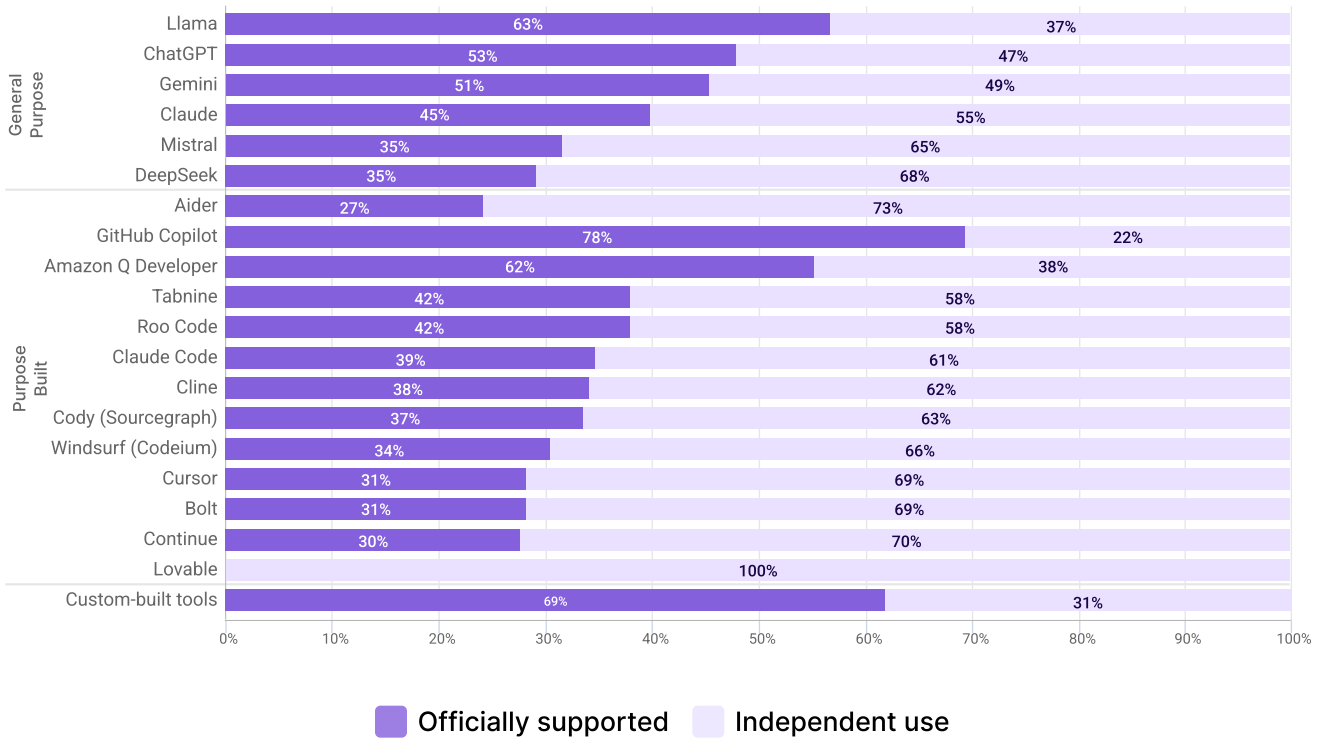


Figure 6: Total usage vs. Officially supported usage

The details: Llama was used in 25% of organizations, and it's officially supported in 63% of those instances, more often than any other tool, including ChatGPT (53%) and Gemini (51%). Something like DeepSeek, which is used in 28% of organizations, is officially supported in just 33% of those instances (second lowest percentage of any tool).

GitHub Copilot, which is used in 76% of organizations, is officially supported in 78% of those instances. Amazon Q Developer is supported 62% of the time it's used. Those two options are the defaults available to developers. Other popular tools are used despite a lack of formal support. Claude Code is officially supported just 39% of the time it is used, while Windsurf is at 34% and Cursor is just 31%.

We've heard anecdotally that developers want Cursor, but get Copilot. This data supports that sentiment; and it appears that developers are bringing their preferred tool(s) to work, despite the lack of formal, organizational support.

Section 3: Adoption

Let's now dig a little deeper into adoption patterns. While most respondents' organizations were using multiple tools, the depth of that use varied. Respondents placed their organizations at different points along an adoption journey.

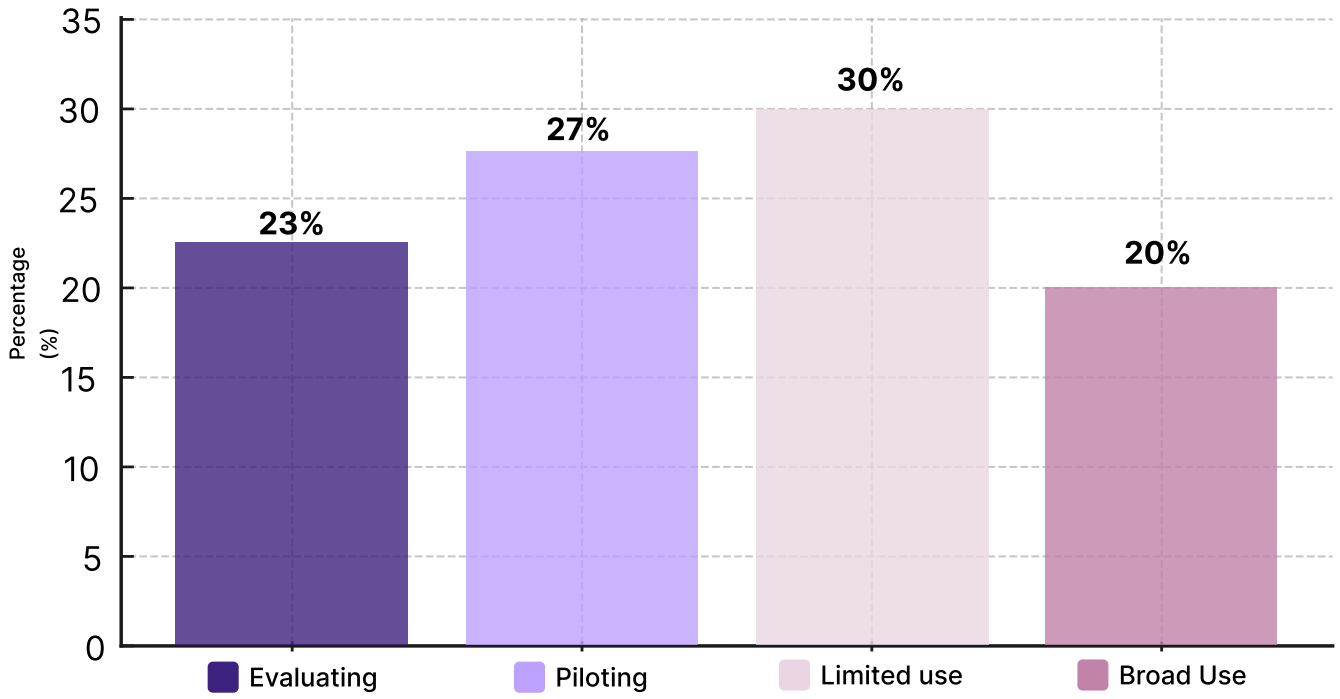


Figure 7: Current depth of AI code generation tool usage

And more than half of respondents (55%) acknowledged that adoption of AI code generation tools is being driven bottom-up.

The intersection of adoption stage and direction highlighted the importance of executive support for AI code generation tools. Of the organizations where tools were in broad use, 77% of the time the initiatives were driven top-down. On the other end of the journey, in organizations that were evaluating tools, 68% of the time the effort was spurred bottom-up. In summary, it may be up to individual developers and teams to get the ball rolling, and to activate leadership when the organization is ready for wider use.

Regardless of level in the org, there was universal bullishness on increasing use of AI code generation tools. 47% said that more than half of their developers use AI code generation tools on a daily basis today. Respondents anticipate that over the next 12 months, that number will grow to 70% of their developers.

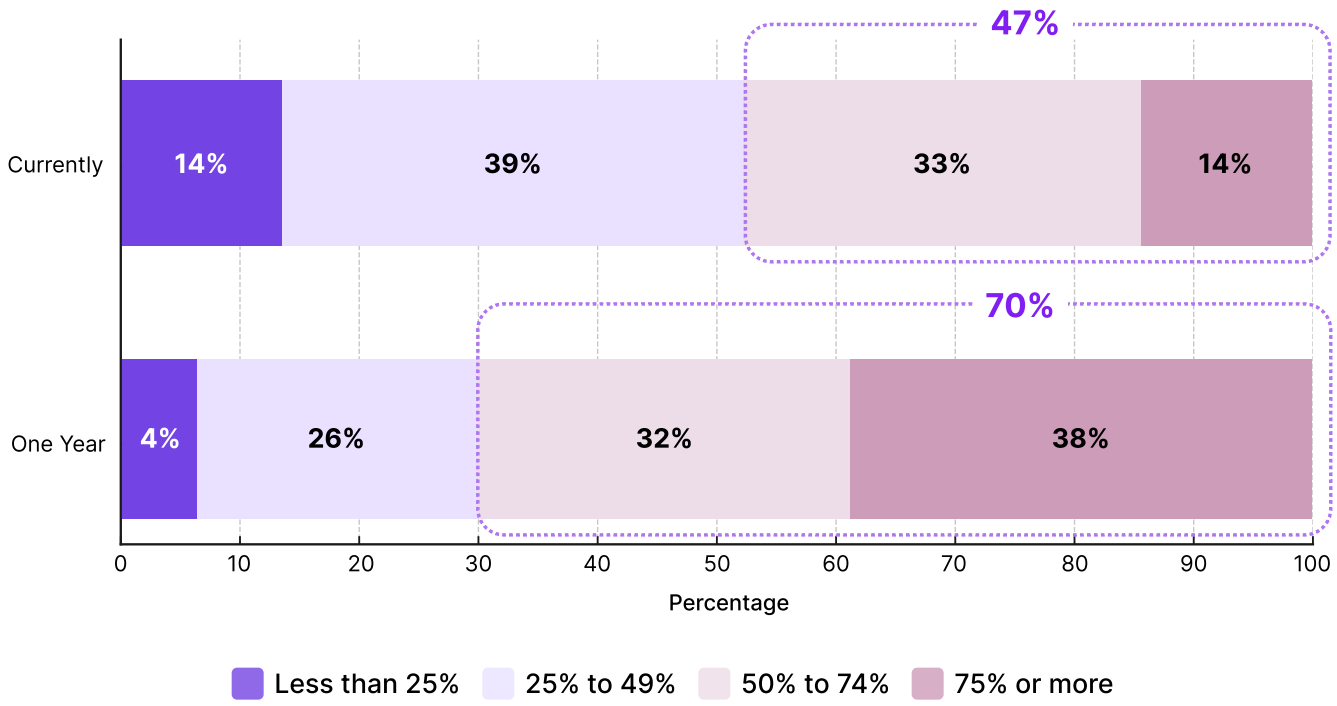


Figure 8: Percentage of developers using tools on a daily basis

We're clearly at a point of increasing dependence: 87% of respondents said that their developers would be angry 😡, frustrated 🤯 or disappointed 😔 if they were no longer permitted to use AI code generation tools. And the higher a respondent was in their org, the more likely they were to perceive that passion; executives were two times more likely to believe developers would be angry if they barred them from using these tools.

Section 4: Benefits

After we explored tool preference and adoption, we asked respondents about the perceived benefits of AI code generation tools. Almost every respondent (98%) said that their organization had benefitted from AI code generation tools. Faster development velocity topped the list, followed by better documentation and developer satisfaction.

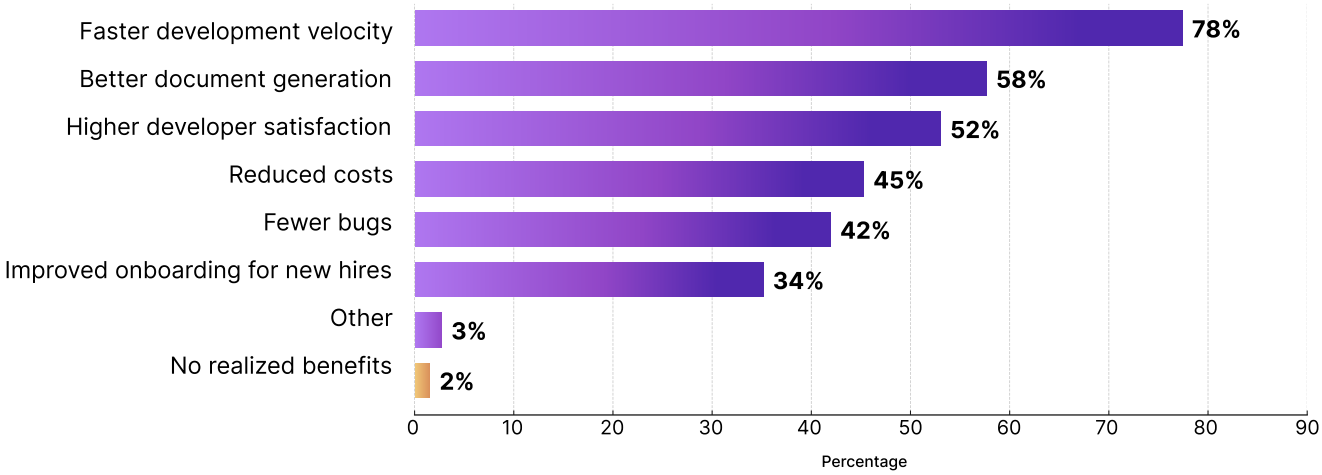


Figure 9: Perceived benefits of AI code generation tools

We decided to take a closer look at how tool choice affected perceived benefits. A primary difference surfaced between respondents using Copilot as compared to those using IDE-based agents, specifically Windsurf and Cursor. Windsurf and Cursor users were more likely to perceive benefits than Copilot users, specifically faster development velocity (85% v. 76%), better documentation (68% v. 53%) and improved onboarding of new hires (51% v. 29%).

Respondents clearly believe their teams are more productive with the help of these tools. And while 75% acknowledged that they “struggle to accurately assess the ROI of AI code generation tools”, we asked for a gut check on relative developer productivity. On average, respondents pegged the increase in developer productivity from AI code generation tools at 51%. But responses varied widely, with 7% seeing the productivity bump as <20% and 13% believing the boost was >80%.

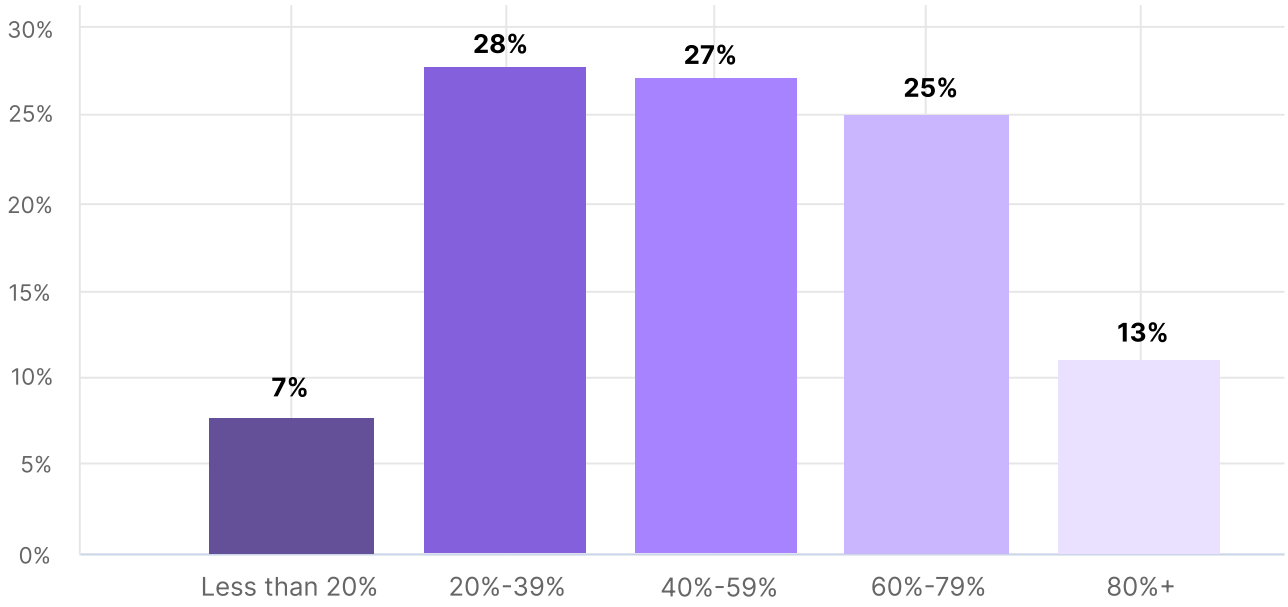


Figure 10: Perceived productivity increase from using AI code generation tools

While these tools are making developers more productive, is that incremental productivity directed at the surface area that matters most to respondents (and presumably, their organizations)? 68% of respondents said that "AI code generation tools would be better applied to brownfield instead of greenfield code". And 75% of respondents agreed that "individual developers would prefer that AI tools handle operational tasks rather than creative code generation tasks."

There's some implied (and understandable) concern that AI code generation tools are stripping away more of the work that drives developer satisfaction. There's a clear desire to set these tools loose on brownfield code to tackle more asynchronous and operational work; but, as we'll see in subsequent sections, respondents don't yet have the necessary governance or trust.

Section 5: Obstacles

It's early innings for AI code generation. And as engineering leaders seek to push their teams and organizations from evaluation and piloting to production use, there are real obstacles. Survey respondents pointed at lack of trust in tool outputs and legal or compliance concerns as the top two impediments.

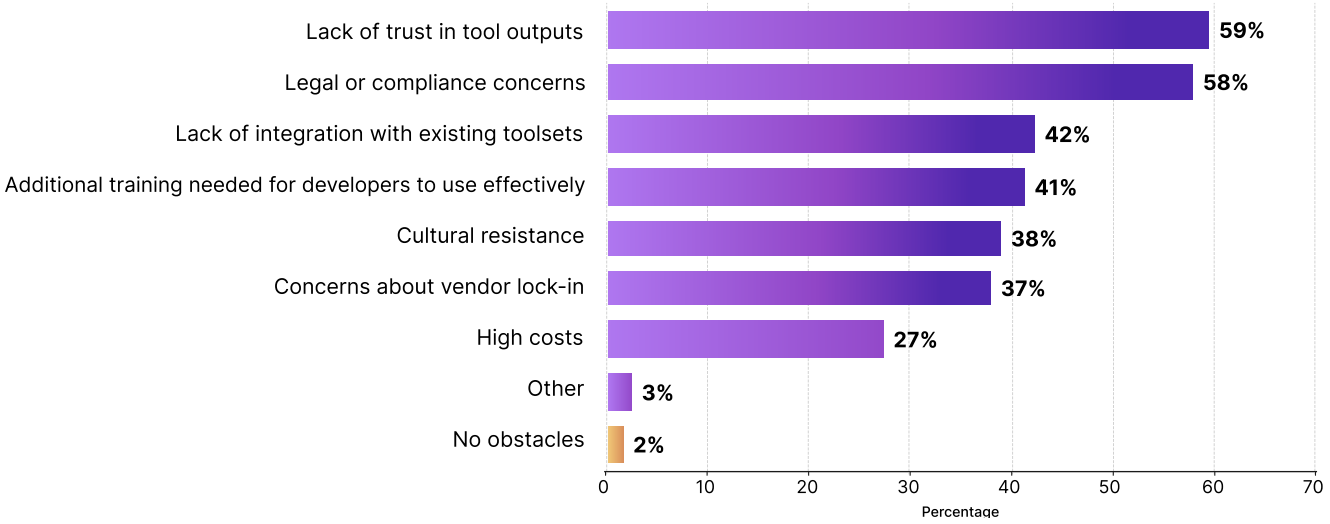


Figure 11: Perceived obstacles to adopting AI code generation tools

Picking up on trust, we asked respondents about their willingness to defend AI-generated code. Only 24% said they would be very confident. 63% sat on the fence, stating they would be somewhat confident. 9% were flat out not confident, and 4% said they'd never use AI-generated code in a situation that mattered.

The lack of trust likely stems from lack of guardrails. While 97% of respondents have some policies in place governing AI code generation, 25% said those policies were informal at best and 49% agreed their policies need work.

A thru-line in the survey were the occasional disconnects between executives and team managers; belief in the existence of governance and policy was a clear example. Specifically, executives were almost twice as likely as team managers (33% v. 17%) to say that their policies were formally documented.

The other missing guardrail is security. Respondents listed security posture and compliance as the most important factor when choosing an AI code generation tool (59%). Integration with an existing toolset was a close second (54%) and all other considerations were a distant third.

One of the ways we can address gaps is by better supporting developer's use and adoption of these tools. So, we closed by asking these engineering leaders what actions they're taking, and responses were varied. Some organizations are using training sessions, others are leaning on internal experts and mentors. Given the pace of change, engineering leaders are trying to keep up.

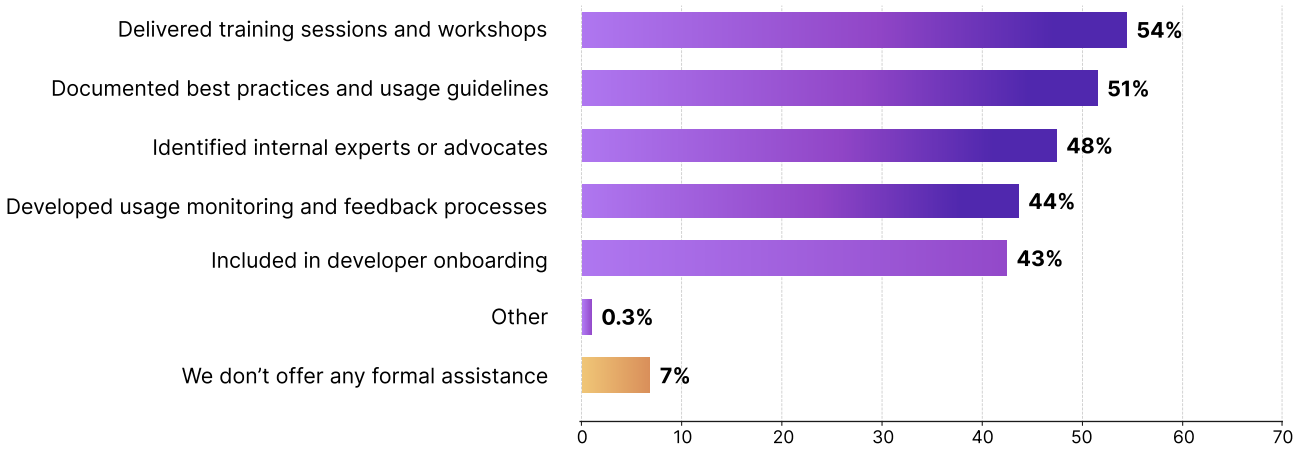


Figure 12: Types of support that organizations provide developers to facilitate tools adoption

Closing

Stepping back from the results, we see a few themes emerge:

- Developers need a bigger toolbox. The number of tools used is growing rapidly; some of those are formally supported (by organizations), but many are not.
- Productivity—security paradox. The speed of innovation (and adoption) is outstripping security protocols and respondents are understandably anxious.
- Dev v. Ops priorities. The most immediate impact of these tools is development velocity. While that translates into productivity gains, respondents see potential to apply these tools to high impact (but complex) brownfield and operational work.
- Role-based reality distortion. The higher a respondent is in their organization, the more critical they are to driving real use, and the more confident they are that the right governance is in place.

It's early for AI code generation, and we're excited to see how engineering leaders' perceptions and priorities evolve. If you're keen like us, then let's talk. Jump into our Discord and let us know what data point struck you. We'd welcome your ideas for questions or topics to probe in future surveys of engineering leaders and / or developers. Talk soon!