

2024

Measuring Impact: The GenAI Code Report

With data sourced from 150+ CTOs and VPs of Engineering, this LinearB Research Report breaks down how to measure the impact of Generative AI code across the software delivery lifecycle.

TABLE OF CONTENTS

The Dawn of Generative AI Code	01	AI Metrics	11
GenAI Adoption	02	How LinearB Does It	13
GenAI Benefits	03	The Future of Initiative Tracking	15
GenAI Risks	04	Appendix	16
Survey Results	05		

The Dawn of Generative AI Code

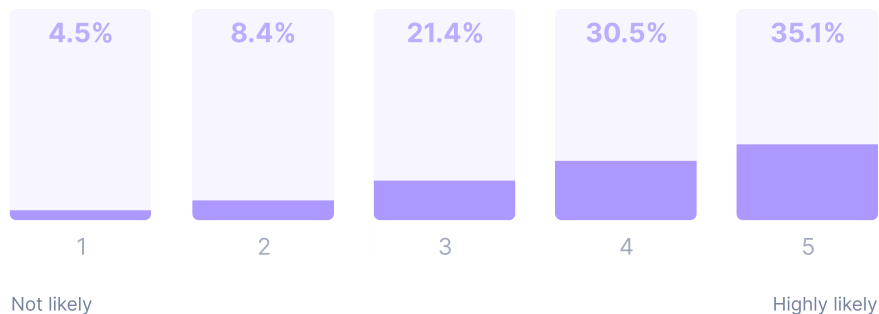
By the end of 2024, Generative AI is projected to be responsible for generating **20% of all code** – or 1 in every 5 code lines.

Nearly every engineering team is thinking about how to implement GenAI into their processes, with many already investing in tools like GitHub Copilot, Amazon CodeWhisperer, Codium.ai, Tabnine or similar to kickstart this initiative.

In fact, our survey revealed that 87% of participants are likely or highly likely to invest in a GenAI coding tool in 2024.

How likely is your organization to invest in a Generative AI coding tool in 2024?

3.8/5 Average rating



As with any new tech rollout, the next question quickly becomes, “How do we measure the impact of this investment?” It’s every engineering leader’s responsibility to their board, executive team, and developers alike to zero in on an answer and report their findings.

At LinearB, we break this question of impact into the following three categories:

 Adoption

 Benefits

 Risks

In this year’s GenAI Code Report, we break down each of these categories in detail, sharing key insights from our survey, and explaining how to track metrics that measure the impact GenAI has on the software delivery process.




The following data was compiled from a study of 150+ CTOs, VPs of Engineering, and Engineering Managers – hailing from start-ups and Fortune 500 companies alike.

We hope that you’ll walk away from this report not only understanding how you can start measuring the impact of GenAI code, but also the mindset of your peers moving forward.

GenAI Adoption




Many engineering leaders are managing dozens, if not hundreds, of developers, making it nearly impossible to manually monitor usage of any new tool, let alone one as all-encompassing as GenAI.

Once you've started using a product like Copilot – or even if you're still considering rolling one out – there are three main questions you'll want to answer:

-  What use cases are we solving with GenAI?
-  Which parts of my codebase are being written by a machine? New code? Tests?
-  Is my team actually using this? If yes, to what extent? Are we scaling our usage over time?

As a baseline, tracking the number of **Pull Requests (PRs) Opened** that have GenAI generated code is a great place to start measuring adoption.




By studying this metric across various segments of your engineering org – and across different timelines – you can answer more advanced questions about adoption like:

-  Which devs, teams or groups are faster to adopt GenAI?
-  Is adoption still growing, has it plateaued or is it tapering off after initial excitement?
-  Which parts of the codebase (e.g. repositories, services, etc.) are seeing the highest adoption?

GenAI Benefits

Then, there's the question of benefits – is GenAI actually helping my team the way I expected it to?

In theory, a GenAI tool that's writing chunks of code for your team should be reducing coding time and speeding up simple tasks. You'll want to evaluate your team's throughput and velocity to answer the following questions:

-  Are we getting the intended benefits from GenAI?
-  Is my team moving faster as a result?
-  Are we actually shipping more value?

There are several metrics you can track here:

Coding Time

The time it takes from first commit until a PR is issued. This metric can help you understand whether GenAI is actually helping to reduce the amount of time it's taking developers to code.

Merge Frequency

How often developers are able to get their code merged to the codebase. Merge frequency captures not just coding time, but also the code review dynamic. This metric can help you understand whether GenAI is helping your developers move faster – a higher merge frequency indicates faster cycles.

In addition, use **Completed Stories** to track an increase in story delivery velocity, and **Planning Accuracy** to track improved predictability in your sprints – two possible, though indirect, benefits of introducing GenAI into your coding process.

“Software projects can be unpredictable due to a multitude of reasons - from unforeseen technical challenges to scope changes. Engineering metrics, such as Planning Accuracy and workflow automation tools have helped us increase predictability in release schedules and timelines.”



Marko T.
CTO



⚠️ GenAI Risks

Along with GenAI adoption and benefits, there are some important risks to track. Besides common concerns about security, compliance and intellectual property, the adoption of GenAI for coding heralds a fundamental change in the “shape” of your code delivery and how your codebase gets updated.

While GenAI allows for much faster creation of code, are your review and delivery pipelines ready to handle this? It is essential that you're able to gauge how your teams' processes are being impacted so you can get ahead of any risks before they affect delivery.

Now, let's get into some core metrics to track here.

PR Size

GenAI makes it easy to generate code. An inflation in average PR size is an early indicator of inefficiencies, as larger PRs are harder and slower to review, and far riskier to merge and deploy. Track PR size to ensure your teams' GenAI PRs aren't bloating.

In many cases, generated code is more difficult to review. It's harder for the PR author to reason about it and defend it - since they didn't actually write the code themselves.

Use the **Review Depth** and **PRs Merged Without Review** metrics to ensure GenAI PRs get proper inspection, and the **Time to Approve** and **Review Time** metrics to check for bottlenecks in your development pipeline.

Finally, use **Rework Rate** to understand if there is an increase in code churn, and **Escaped Bugs** to track overall quality trends.

"Tracking engineering metrics has helped me as the VP of R&D to have educated discussions with my direct reports and with my CEO. I can identify bottlenecks quickly, measure team efficiency and the developer experience, then improve based on data."



Idan Lavy
VP of R&D

SOLIDUS
LABS



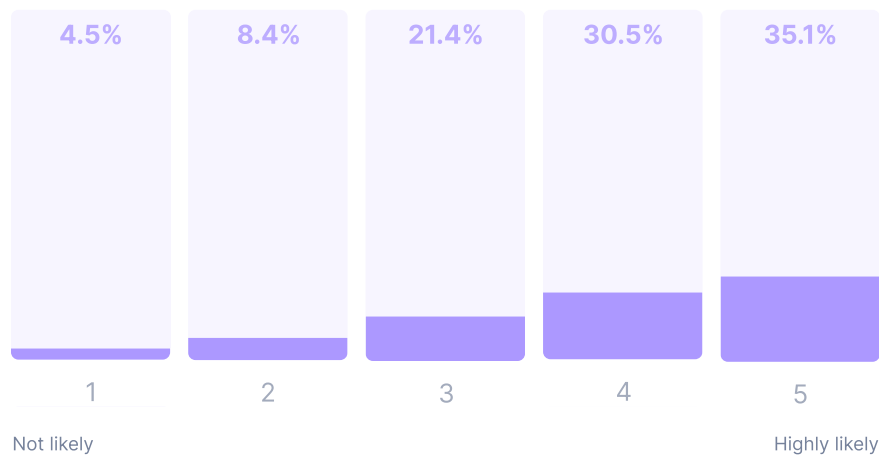
SURVEY RESULTS

GenAI Adoption State

87% of Orgs are planning to invest in a GenAI tool in 2024.
71.6% of Orgs are currently in or through a GenAI tool adoption process.

How likely is your organization to invest in a Generative AI coding tool in 2024?

3.8 Average rating



Please describe your current status in regards to a Generative AI coding tool adoption:



The writing is on the wall: GenAI is poised to revolutionize the software delivery landscape as we know it, with massive implications around the way developers will write, test and ship code for years to come.

Perceived Use Case by Respondent Seniority

Managers are focused on outcome-related use cases for GenAI Tools

- Code Reviews
- Documentation

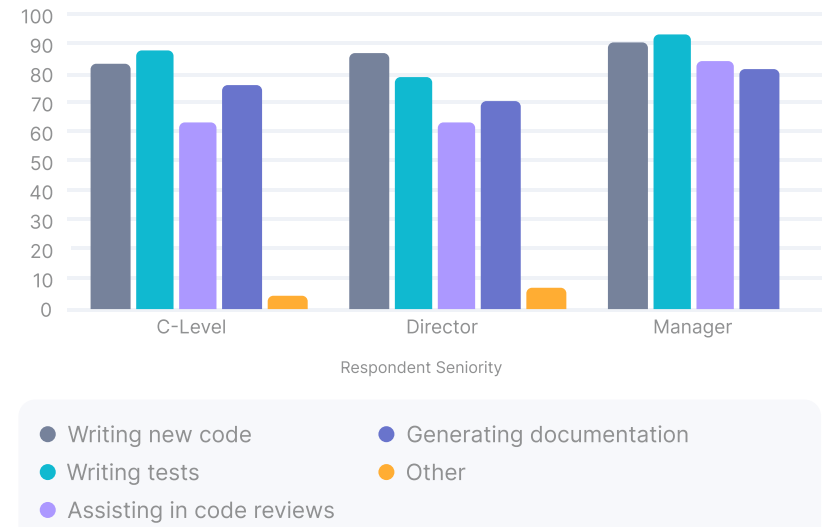
Executives are focused on productivity and quality-related use cases for GenAI Tools

- Writing Code
- Writing Tests

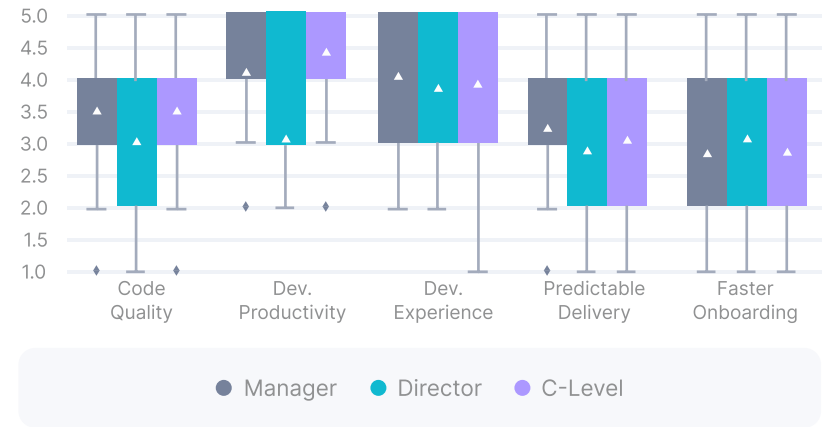
In our data, we noticed a positive correlation between company size and direct measurement as a means of assessing GenAI impact.

This implies larger companies feel more capable of setting a metrics program in place.

What are your most likely use cases for a Generative AI coding tool?



Rate the improvement you expect to see from adopting a Generative AI coding tool

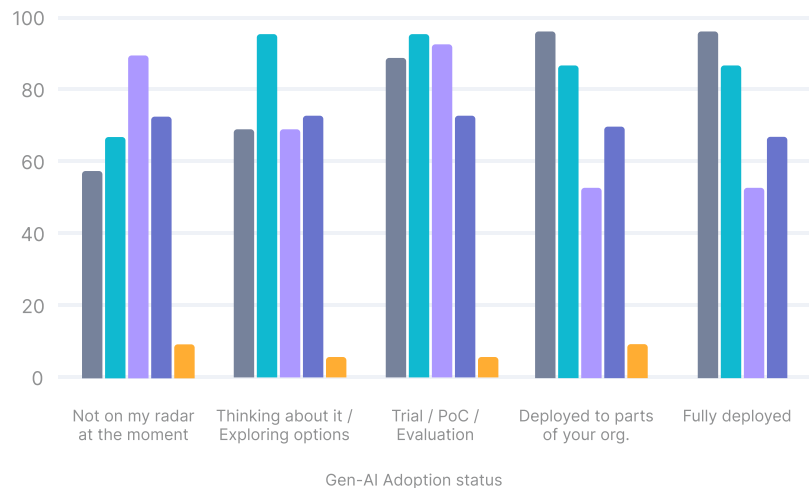


Use Case by Adoption

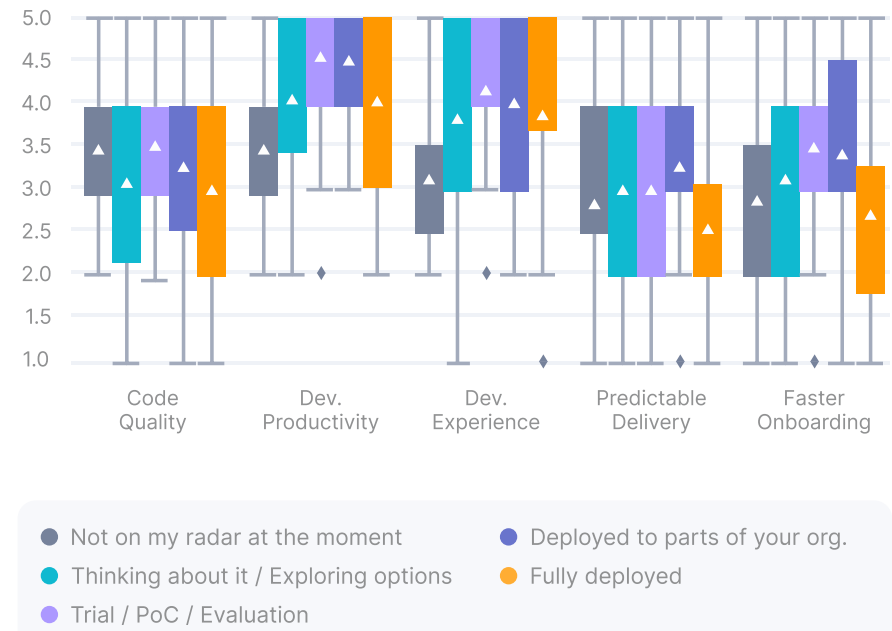
Organizations who are further along in their GenAI tool adoption journey see **Developer Productivity** and **Developer Experience** as their key use cases.

Code Quality and **Assisting in Code Reviews** as use cases lose luster as engineering orgs move further down their adoption journeys.

Use Case Importance Breakdown by Adoption Phase



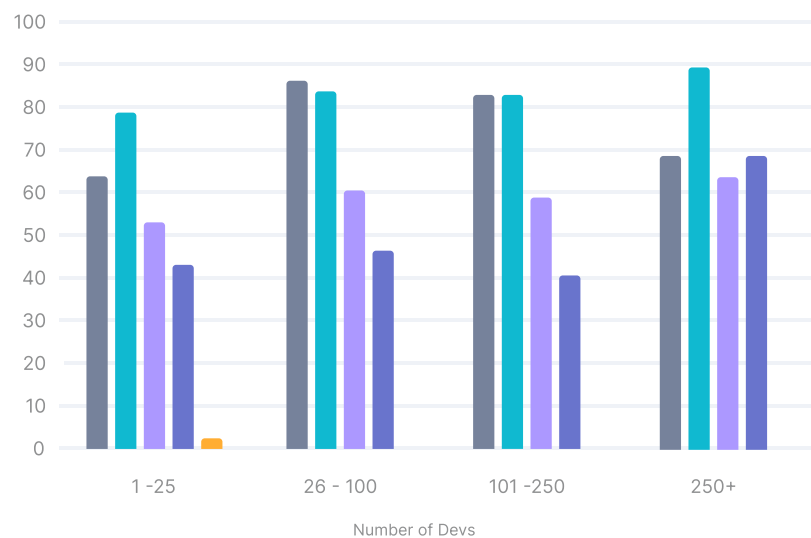
Use Case Importance Breakdown by Adoption Phase



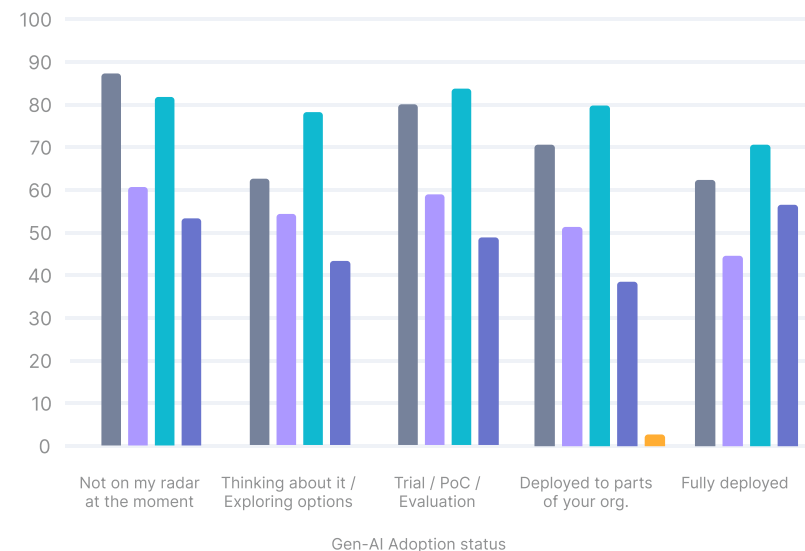
Measuring Impact

For measuring impact, our results revealed that both larger companies and organizations further in their adoption journey prefer direct (hard) metrics to assess the efficacy of their GenAI Code tools vs. qualitative (soft) surveys.

Impact Measurement Choice by Dev Org Size



Impact Measurement Choice by Adoption Phase

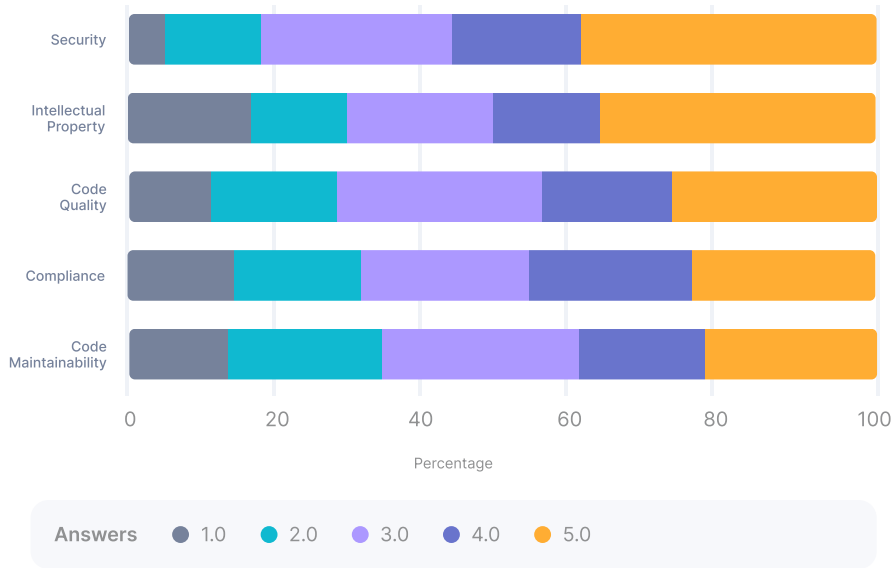


Risks: Perceived vs. Reality

Security is the top concern for all – followed by Compliance / Quality / Intellectual Property Concerns.

Concern levels drop across the board as adoption grows, with IP and Compliance taking the biggest hits.

Score Distribution Across Concern Areas



Concern Levels by Adoption Phase



- Not on my radar at the moment
- Thinking about it / Exploring options
- Trial / PoC / Evaluation
- Deployed to parts of your org.
- Fully deployed

AI METRICS

AI Metrics

In this section, we'll define the most important AI impact metrics broken down by adoption, benefits, and risks – and then we'll talk you through how our team is measuring them today.

With LinearB, engineering leaders can track a custom dashboard with the metrics below, so they can evaluate the status of their GenAI initiative at any given point in time, from rollout to full utilization.

Adoption

Pull Requests Opened

A count of all the pull requests issued by a team or individual in all the repositories scanned by LinearB, during a selected time frame.

Pull Requests Merged

A count of all the pull requests merged by a team or individual in all the repositories scanned by LinearB, during a selected time frame.

Benefits

Merge Frequency

The average number of pull or merge requests merged by one developer in one week. Elite merge frequency represents few obstacles and a good developer experience.

Coding Time

The time it takes from the first commit until a pull request is issued. Short coding time correlates to low WIP, small PR size and clear requirements.

Completed Stories

A sum of the story points that transitioned to and remained in the "Done" state through the end of a given time period or sprint.

Planning Accuracy

The ratio between planned issues and what was actually delivered from that list. Aim for 70% as a benchmark to start improving planning accuracy.

Risks

PR Size

The number of code lines modified in a pull request. Smaller pull requests are easier to review, safer to merge, and correlate to a lower cycle time.

Rework Rate

The amount of changes made to code that is less than 21 days old. High rework rates signal code churn and is a leading indicator of quality issues.

Review Depth

The average number of comments per pull request review.

PRs Merged Without Review

All the pull requests that were either merged with no review at all, or pull requests that were self reviewed.

Time to Approve

The time from review beginning to pull request first approval.

Review Time

The time from the first comment made on a pull request until it is merged.

How LinearB Does It

LinearB's approach to measuring adoption, benefits and risks for GenAI code starts with PR labels. Every pull request that includes GenAI code is labeled, allowing metric tracking for this type of work. From there, you can compare success metrics against the unlabeled PRs.




Engineering leaders have two main options when it comes to tracking PRs with Gen AI code:

- Manual Labeling
- Workflow Automation with **gitStream**

The first option, albeit time-consuming, is certainly doable. To start, ask your team to manually label all PRs that they authored with GenAI. Or you can configure a team in LinearB of all contributors that have access to GenAI tools.

From there, you can conduct an analysis that compares quality and efficiency metrics from before and after your teams' GenAI usage started (more on this later).

Alternatively, you can leverage a workflow automation tool, like gitStream, to auto-label PRs based on criteria such as:

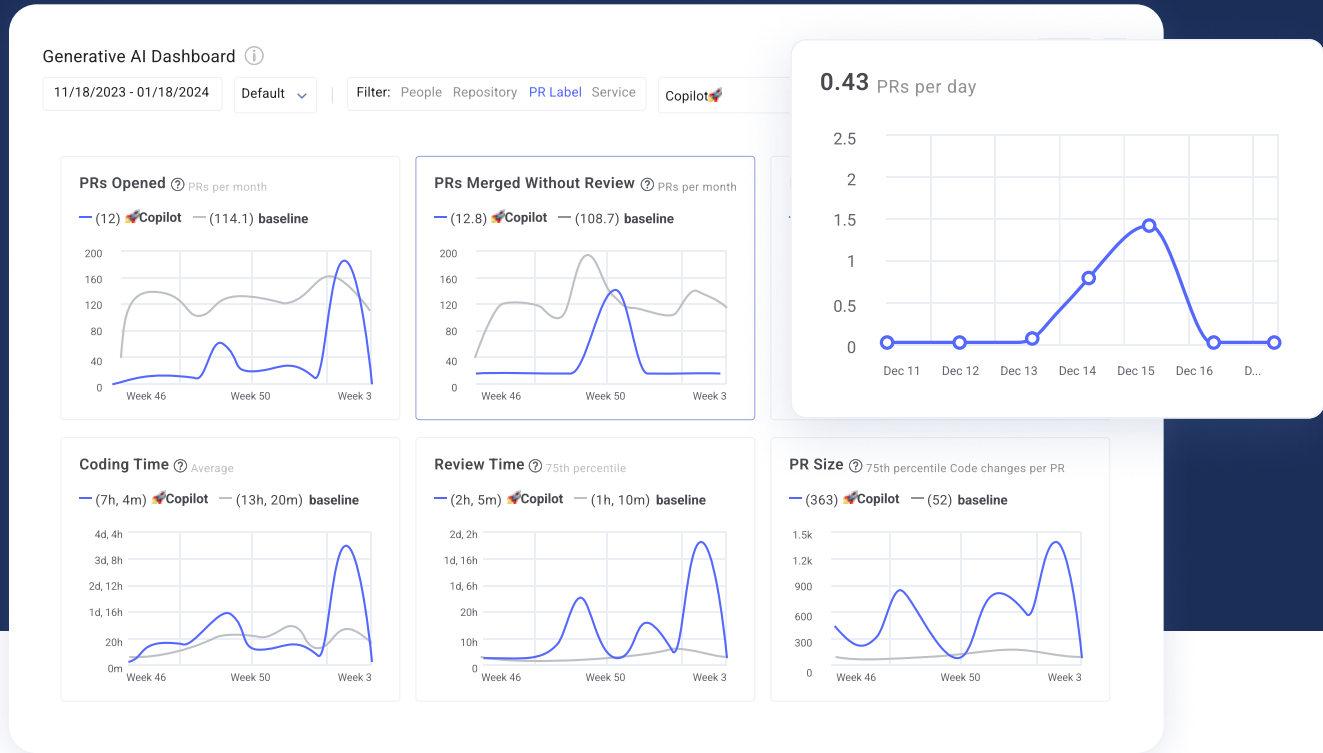
-  **List of users** included in the GenAI experiment
-  **Hint text** in the PR title, commit messages, or PR comments
-  **A required Yes/No button** for devs to indicate whether or not they used GenAI to assist coding

"Workflow automation has had a cascading impact on improving business outcomes while enabling my team to focus on solving the correct problems. Tracking delivery metrics helped us pinpoint areas of flow that could be enhanced, dramatically boosting team morale and engagement."



Craig W.
Head of Engineering





How LinearB Does It

This is our custom GenAI dashboard. Note that we've labeled all PRs with a Copilot label in GitHub, and used this PR label in the dashboard filter definition. In this view, the blue line at the bottom represents all PRs with a Copilot label, and the gray line represents all other PRs (our baseline in this case).

Now, let's take a closer look at PRs merged without review, for example. At LinearB, our engineering team has collectively agreed that no PR written using GenAI should ever be merged without review.

In the graph, you can see that we had a spike in PRs merged without review in Week 50. This is an anomaly that we'd want to investigate further. Whenever we see a trend in our data that doesn't match our expectations, we drill deeper into our data by clicking on the spike in the graph. This will bring up a list of all the PRs that contributed to this datapoint, so we can take a closer look at all the GenAI PRs we merged without review.

Moving forward, we can also leverage gitStream to block all GenAI code from getting merged without a proper review.

The Future of Initiative Tracking

As exciting as it is to talk about the future of the software delivery landscape, the reality is that GenAI is likely just one initiative that you as an engineering leader have in flight right now.

Your mind is probably also spinning about your developer experience initiative, your agile coaching initiative, your merge standards initiative, your test coverage initiative, your new CI pipeline initiative, etc, etc.

Universal label tracking with [gitStream](#) and [LinearB](#) allows you to measure the impact of any initiative you've kicked off with your team. This way, you can answer questions like:



What is the ROI on this new 3rd party tool we bought?



Should we roll this agile coaching initiative out to the rest of the organization?



Is changing up our CI pipeline allowing us to speed up our delivery? By how much?

Advocating for more headcount or an increased budget is much more effective when you can point to a dashboard with tangible engineering results that you can relate to any given initiative, GenAI or otherwise.

? Did You Know

LinearB metrics and workflow automation have already saved developers thousands of hours, with the average repo seeing a 61% decrease in Cycle Time.

Start tracking the impact of your GenAI initiative today **with a free forever account!** If you'd like to discuss any of what was covered in this report in more detail, or you want to see some of the more advanced features, **schedule a demo.**

APPENDIX

Does your engineering organization currently use Generative AI tools to produce or contribute to new code creation?

Github Copilot



CodeWhisperer



Codium.ai



Other



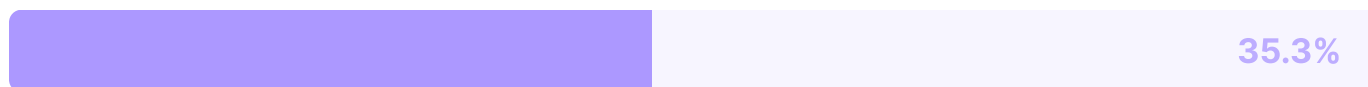
Who took the survey?

Location

EMEA (Europe, Middle East, Africa)



NA (North America)



APAC (Asia, Pacific)



LATAM (Latin America)



Who took the survey?

Department

Engineering



IT



Product



Other



Who took the survey?

Level

Manager



Director



C-Level



Other



Who took the survey?

of devs at your company

1 - 25



26 - 100



101 - 250



250 - 1000



1000+

