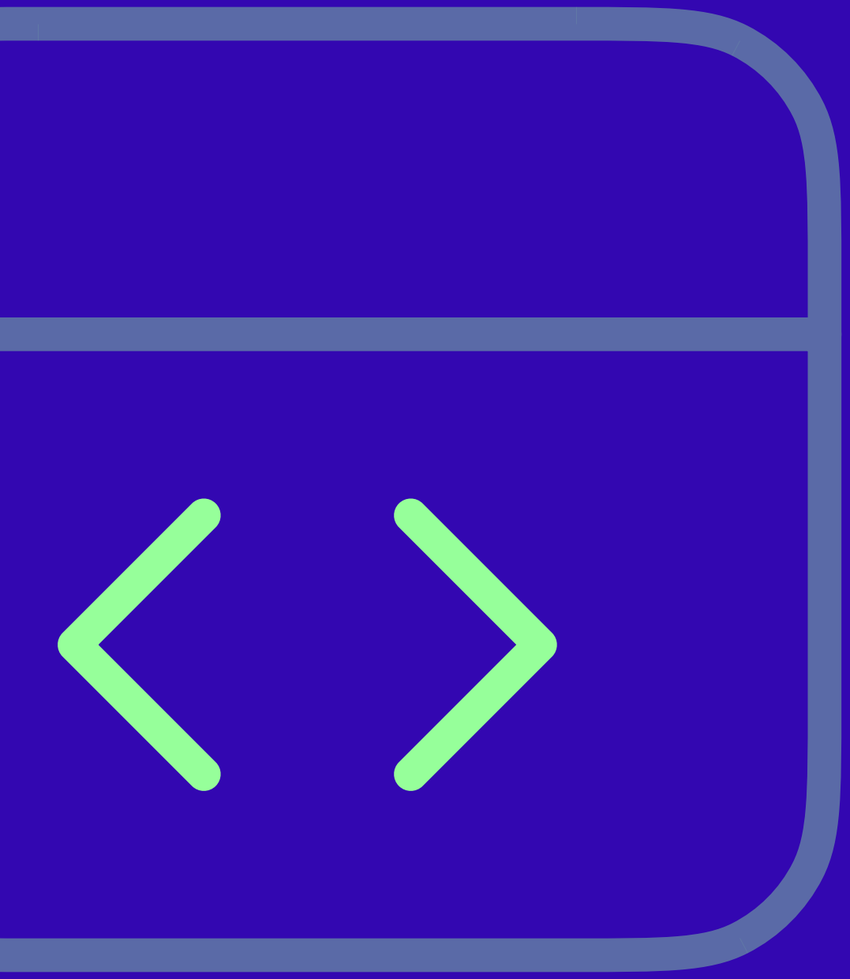
THE 2024 DOCKER STATE OF APPLICATION DEVELOPMENT REPORT

TABLE OF CONTENTS

EXECUTIVE SUMMARY



EXECUTIVE SUMMARY

The 2024 Docker State of Application Development Report provides a deep-focus snapshot of today's rapidly evolving world of software development. Conducted by Docker's User Research Team, the second annual survey asked more than 1,300 respondents a wide-ranging set of questions about their work — from what tools they use to their processes and frustrations, feelings about industry trends, participation in developer communities, and Docker usage. We wanted to know what our community is focused on, what they're working on, and what's most important to them.

2024 REPORT KEY FINDINGS

29%

transitioning to microservices

The rise of microservices continues unabated, with nearly three times more respondents saying they were transitioning from monolithic to microservices than were moving in the other direction, from microservices to monolithic (11%).

36%

report their main development environment is remote, pointing to the growing popularity of developing software in the cloud.

64%

use AI for tasks such as writing code, documentation, and research, and 46% are working on ML in some capacity, underscoring AI/ML's penetration into the software development field.

Although only 14% of respondents view shifting security to the left as an important industry trend, many say they want better security tools, suggesting the shift-left approach to security is a source of frustration and an area where more effective tools could make a difference.

Respondents hit sticking points in project-level tasks prior to development (planning, estimation, and design). However, they also see room for improvement in the development process itself: 20% report they get stuck during debugging/ troubleshooting or testing, and testing is one of the most-selected areas where respondents want better tools.

RESEARCH METHODS

Reflecting the changing state of the industry, this year's report drills down into three main areas:

The state of application development today

AI's expanding role in application development

Security in application development

The online survey is a key vector by which Docker product managers, engineers, and designers gather insights from users to continuously develop and improve the company's suite of tools. This latest report was conducted in late 2023. Its findings are based on 885 completed responses from the roughly 1,300 respondents surveyed.

"The key to world-class application development is knowing how to help software development teams maximize their productivity, get the most out of the disruptive, novel technologies at their disposal, and have a great experience while doing so. The findings in this report demonstrate how Docker continuously seeks to address market needs so that we can better empower development teams not just to compete, but to thrive and innovate with the right processes and tools for their workflows."

Nahid Samsami, Vice President of Developer Experience at Docker

THE STATE OF APPLICATION DEVELOPMENT TODAY

LEARNING TRENDS IN APP DEVELOPMENT

A key engine of developer productivity is effective learning, which entails keeping up with new technologies. We asked respondents how they like to learn, how they find out about new tools, how they learned to code, and what resources they used for online courses and certifications. We also asked about their contributions to open source.

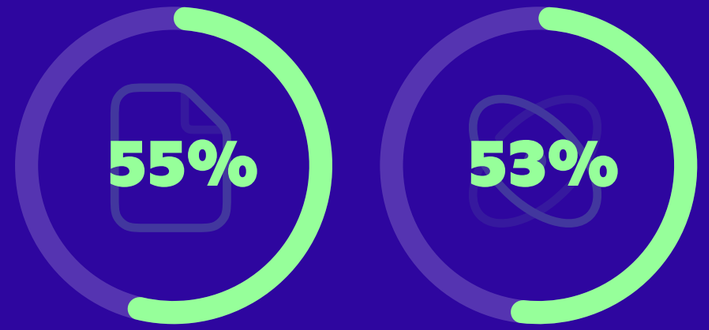
In terms of how they like to learn, respondents favored primary sources and hands-on methods. Most respondents preferred reading documentation (55%) or experimentation/building a side project (53%). Slightly fewer preferred online training courses (47%) and reading source code (45%).

"Industry professionals are tasked with keeping up with an ever more complex and rapidly changing field. To provide them the best experience, it's vital to understand the learning approaches and preferences that support their ability to master new tools and technologies in a way that reduces burden."

Julia Wilson, UX Researcher at Docker

KEY TAKEAWAYS

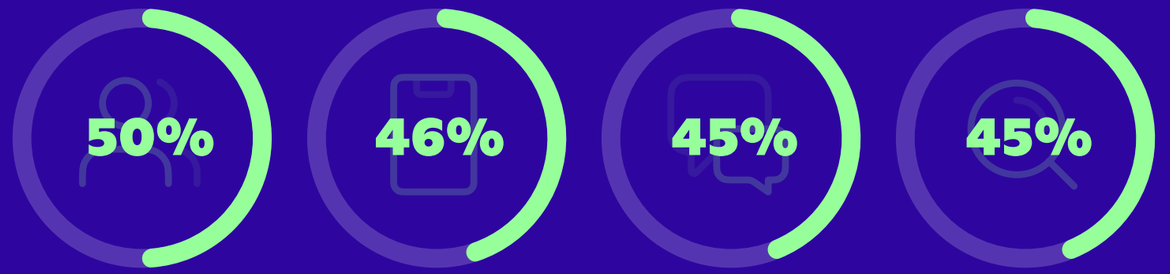
TOP LEARNING PREFERENCES



Reading documentation

Experimentation/building a side project

RESPONDENTS LEARN ABOUT NEW DEVELOPER TOOLS



Developer communities

Social media

Blogs

Searching

50%+ Respondents reported that they learned to code via online resources and/or in school.

RESPONDENTS FIND OUT ABOUT NEW TOOLS LARGELY BY TAPPING THEIR ONLINE AND IN-PERSON NETWORKS

HOW DEVELOPER TOOLS ARE DISCOVERED

50% Developer communities

46% Social media

45% Blogs

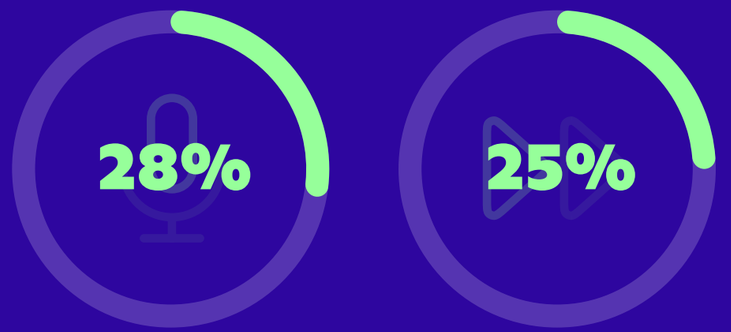
45% Searching

40% Friends and colleagues

Note: In questions with multi-select answers, response percentages do not add up to 100.

DISCOVERY THROUGH EVENTS

RESPONDENTS ALSO LEARN ABOUT NEW DEVELOPER TOOLS THROUGH EVENTS



Conferences

Live conference talks

WHEN LEARNING TO CODE, RESPONDENTS RELIED ON BOTH TRADITIONAL AND ONLINE CHANNELS

LEARNING RESOURCES

54% via online resources (such as videos, blogs, and forums)

45% via online courses or certifications

57%

of industry professionals picked up their coding skills from school

ONLINE COURSES & CERTIFICATIONS

OF THE RESPONDENTS WHO SELECTED ONLINE COURSES OR CERTIFICATIONS

64% of respondents use Udemy

49% of respondents use Coursera

41% of respondents use freeCodeCamp

OPEN SOURCE CONTRIBUTIONS

Many industry professionals contribute to open source projects, whether for professional growth, to give back to the community, or because they get paid to do so. We asked respondents if they contributed — or were interested in contributing — to open source, as well as if their employer allowed them to do so as part of their job.

Fifty-nine percent of respondents said they contributed to open source in the past year, while 41% said they did not. Of the 41% who did not contribute, a large majority (72%) expressed interest in contributing to open source, while less than 25% did not. Clearly, open source software is important to developer ecosystems and communities, with many showing interest even if they don't contribute.

KEY TAKEAWAYS

59%

contributed to open source in the past year

VS

41%

said they did not

OPEN SOURCE CONTRIBUTIONS

Most employers appeared supportive of open source, with more than half of respondents (57%) saying their employer allowed them to contribute to open source as part of their job, and just 20% saying their employer did not.

What keeps developers from contributing to open source? According to respondents, the single biggest obstacle is time — by a long shot (40%). Other barriers included not knowing where to start (23%) and needing guidance from others on how to contribute (16%).

OPEN SOURCE AS PART OF THE JOB

MOST EMPLOYERS APPEARED SUPPORTIVE OF OPEN SOURCE

57% say their employer allowed them to contribute to open source as part of their job

20% say that their employer did not

CHALLENGES OF CONTRIBUTING TO OPEN SOURCE

40%

of respondents say time is the single biggest obstacle

23% say it's not knowing where to start

16% need guidance from others on how to contribute

TECH STACK

We asked respondents about the structure of the applications they work on, how they use containers in development, and what development environment and other tools and technologies they predominantly use.

KEY TAKEAWAYS

51% said they worked on microservices-based applications

WHILE ONLY

35% work on monolithic apps

80% use containers in development

WHILE ONLY

17% do not

(Note: 86% of survey respondents were Docker users.)

36% report their main development environment as being remote

INCLUDING

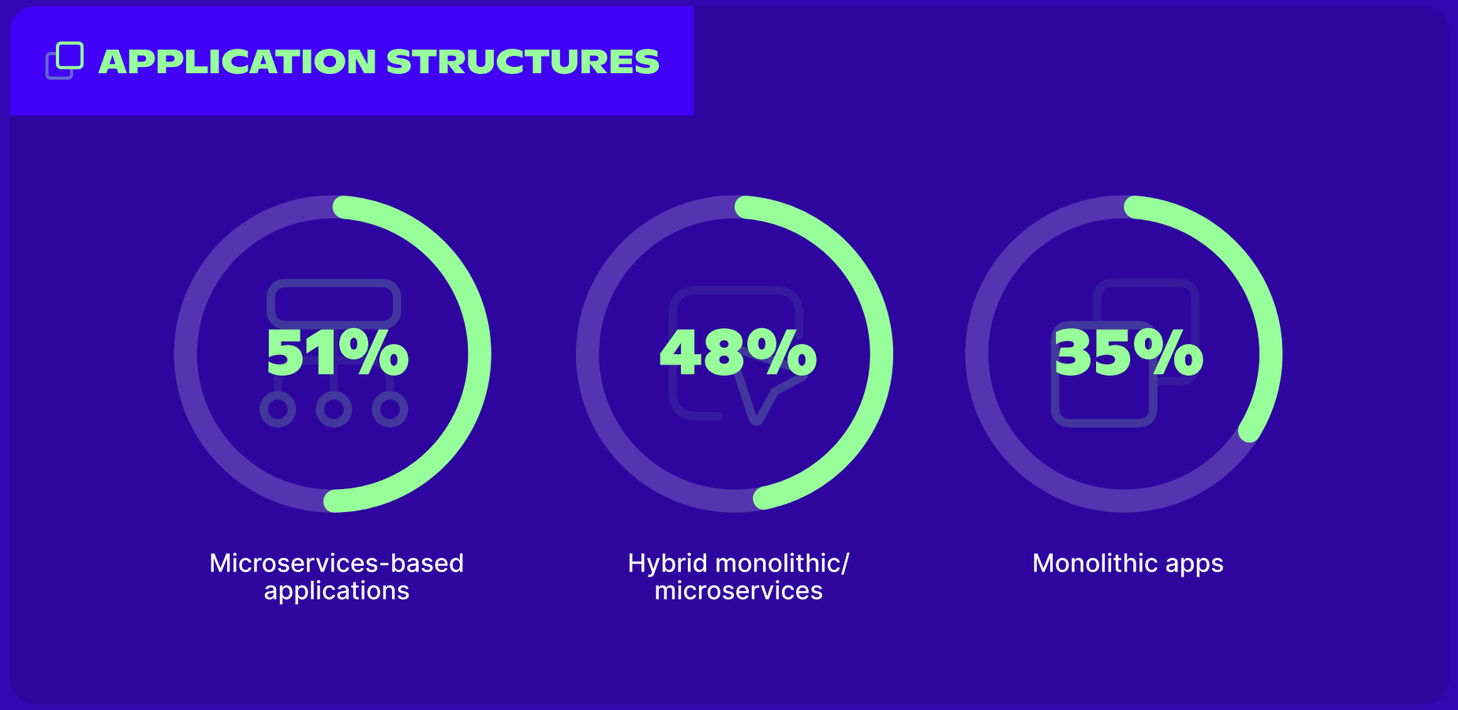
12% who use ephemeral environments

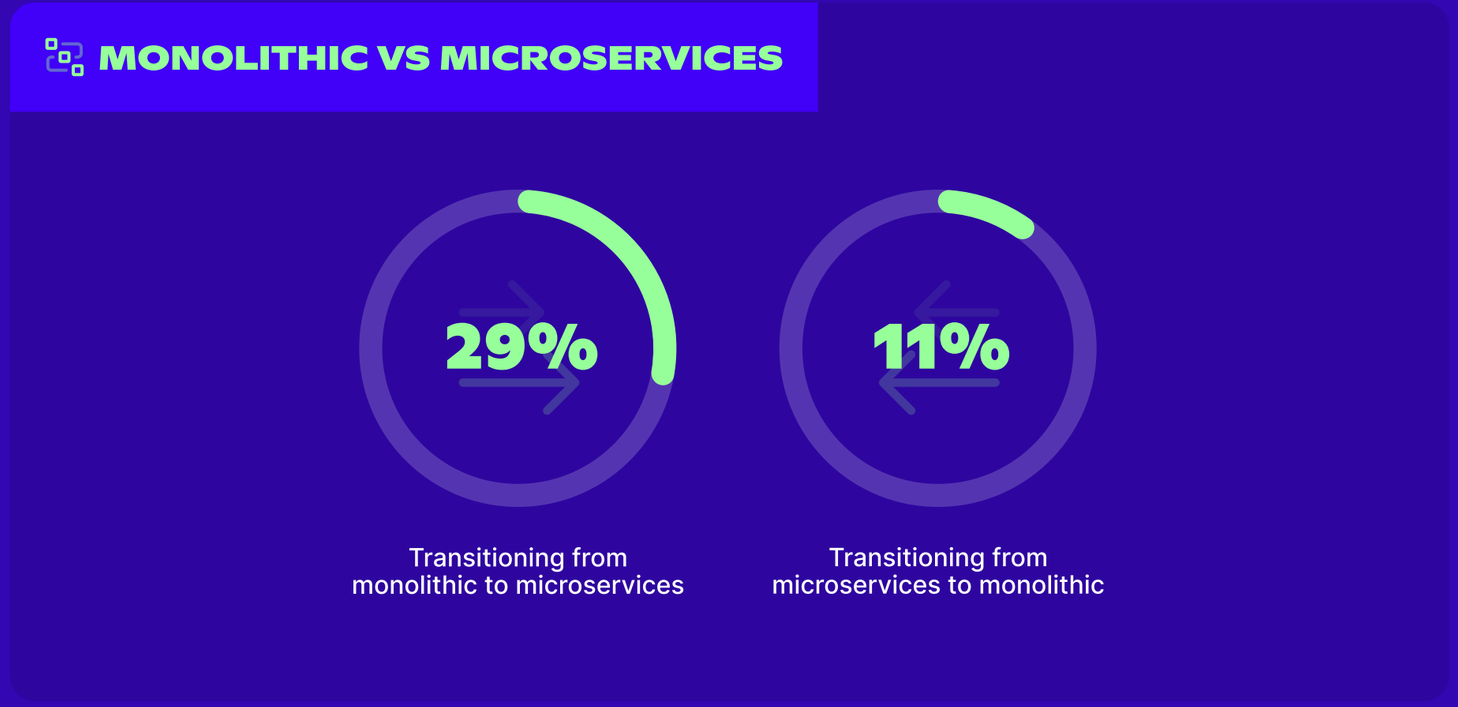
11% who use a personal remote dev environment or cluster

APPLICATION STRUCTURE

Asked about the structure of the applications they work on, respondents' answers underscored the continued rise of microservices.

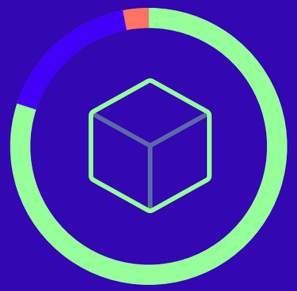
Fifty-one percent said they worked on microservices-based applications, ahead of hybrid monolithic/ microservices (48%) and monolithic apps (35%). Nearly three times more respondents (29%) said they were transitioning from monolithic to microservices than were moving in the other direction, from microservices to monolithic (11%). This suggests that, although microservice systems vary, microservices are swiftly becoming standard in the IT world.





DEVELOPMENT TOOLS OF THE TRADE: CONTAINERS

We asked respondents what they use containers for, and which containerization tools and technologies they use.

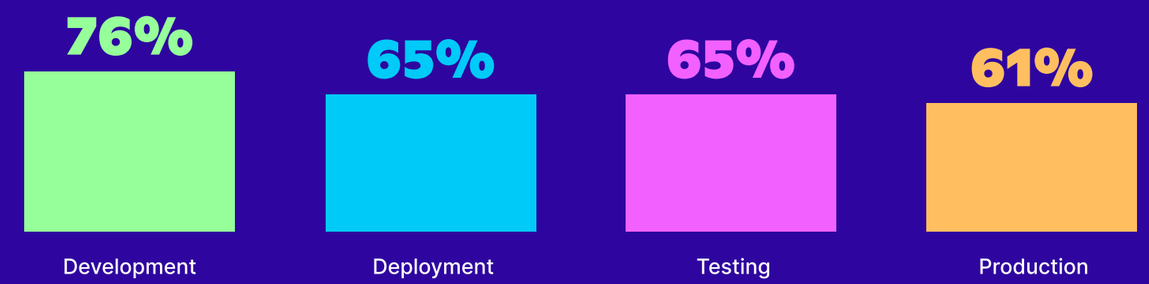


- **80%** Use containers in application development
- **17%** Do not use containers
- **3%** Did not know

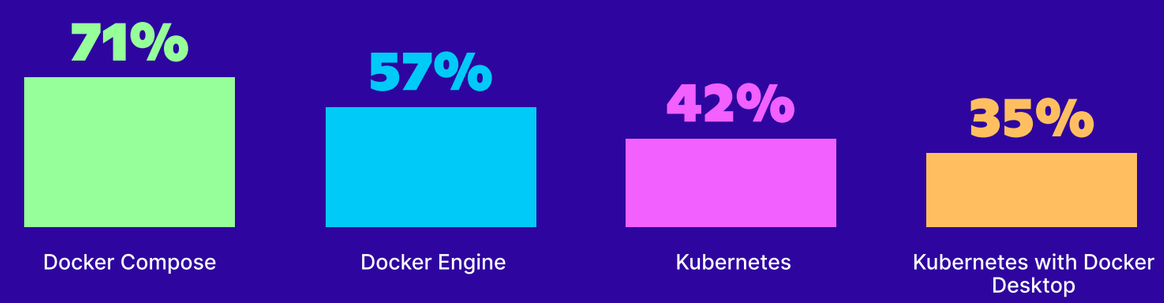
A large majority of respondents (nearly 80%) said they use containers in application development, while only about 17% do not, and about 3% said they did not know. When container users were asked where they primarily use containers, respondents' answers span the lifecycle from development (76%) to deployment and testing (65% each) and production (61%).

Among the broad array of containerization tools and technologies they use, respondents who use containers for development favor Docker Compose (71%), Docker Engine (57%), Kubernetes (42%), and Kubernetes with Docker Desktop (35%).

PRIMARY USE FOR CONTAINERS



CONTAINERIZATION TOOLS AND TECHNOLOGIES



DEVELOPMENT ENVIRONMENTS AND OTHER TOOLS

We asked respondents about their development environment, as well as their preferred operating system, programming languages and frameworks, and data stores.

Although almost 64% reported that their main development environment was still their laptop or desktop, the real story is that over 36% cited remote environments. This included 12% who use ephemeral environments (whether tied to a CI pipeline or not), 11% who use a personal remote dev environment or cluster, and 8% who use remote development tools (such as GitHub Codespaces, Gitpod, Coder, and JetBrains Space).

PREFERRED DEVELOPMENT ENVIRONMENT

64%
Laptop or desktop

36%
Remote environments

REMOTE ENVIRONMENTS USED

12%
Ephemeral environments
(whether tied to a CI pipeline or not)

11%
Personal remote dev environment or cluster

8%
Remote development tools
(such as GitHub Codespaces, Gitpod, Coder, and JetBrains Space)

OVER ONE-THIRD OF RESPONDENTS CITED A REMOTE ENVIRONMENT

This may hint at the growing popularity of developing software in the cloud — a trend fanned by benefits such as increased efficiency, shorter build times, reduced time to market, and faster innovation. The increasing use of ephemeral environments further supports this hypothesis.

Why this apparent increase in reliance on cloud during development? It seems likely that the growing size of applications is a factor, along with the increasing number of dependencies and an overall growth in complexity — all of which would render an all-local environment difficult, if not impossible, to maintain in parity with production.

OS, LANGUAGES, AND DATA STORAGE USAGE

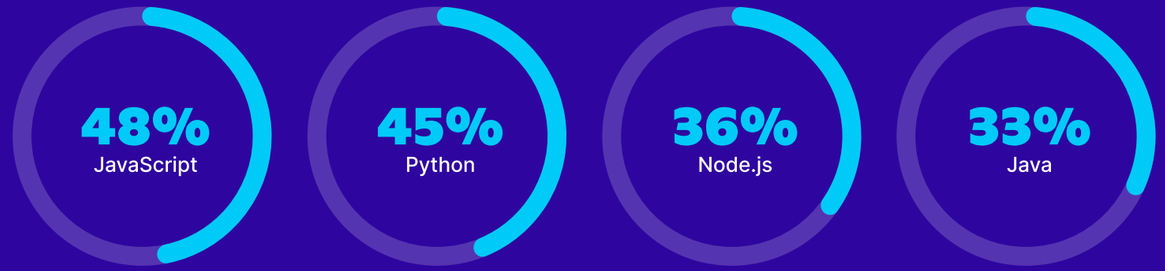
When we asked respondents what operating system(s) they use, Linux (53%) remained the OS of choice for application development, edging out macOS (50%) and ahead of Windows (46%).

There were no surprises in the programming languages and frameworks most beloved by respondents. JavaScript was the most popular (48%), followed by Python (45%), Node.js (36%), and Java (33%).

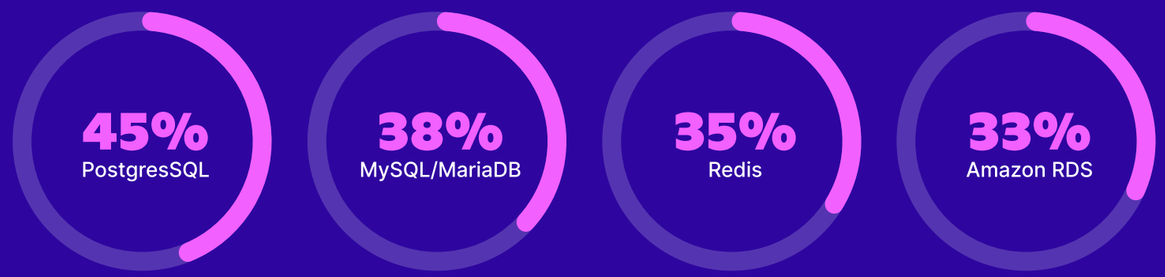
Where do developers store their data? The most popular data stores were PostgresSQL (45%), MySQL/MariaDB (38%), Redis (35%), and Amazon RDS (33%).

OPERATING SYSTEMS

PROGRAMMING LANGUAGES & FRAMEWORKS



DATA STORES



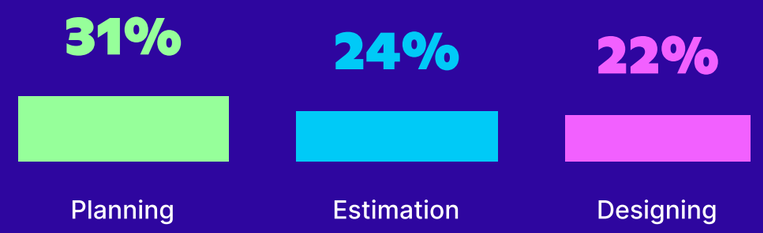
HOW CAN WE IMPROVE THE APP DEVELOPMENT EXPERIENCE?

A primary goal of the survey was to gain insights into how we can improve the app development experience. Respondents were asked where their team gets stuck and where better tools might improve the development process.

We also asked them to rate a variety of tools they use — for development, CI/CD, provisioning, and monitoring. On the process and operations side, we asked them to rate development processes, operational efficiency, and a wide range of tasks at their company or organization.

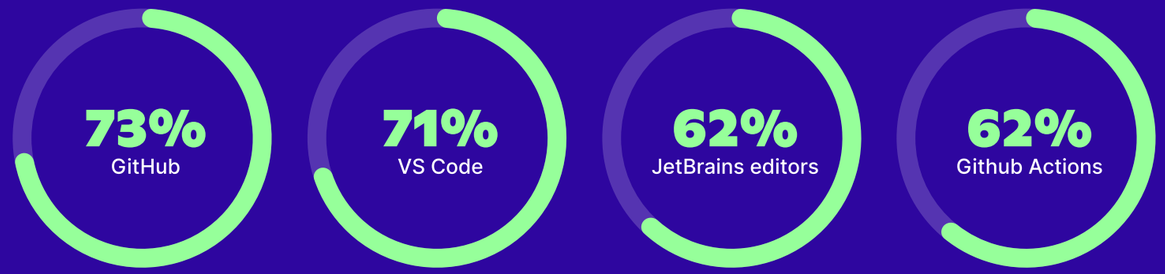
KEY TAKEAWAYS

TOP STICKING POINTS IN THE DEVELOPMENT PROCESS INCLUDE



28% say they need better tools for testing and planning

HIGHEST-RATED TOOLS



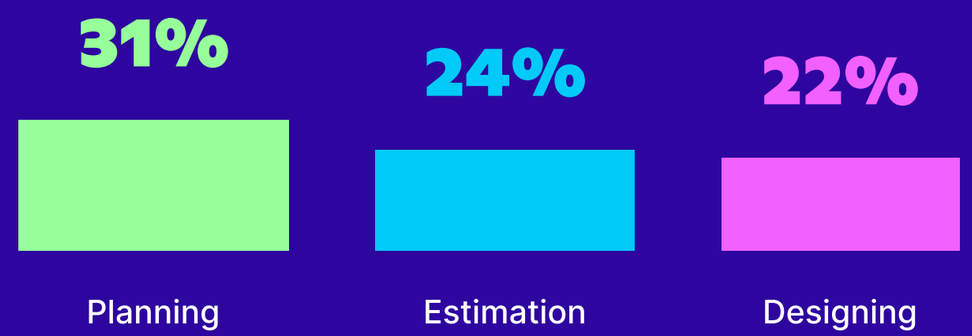
REVEALING ROADBLOCKS IN DEVELOPMENT

Yes, development teams hit walls. When we asked where their team gets stuck in the development process, respondents cited multiple stages, including planning (31%), estimation (24%), and designing (22%).

Planning was also one of the most-selected areas in which respondents desired better tools (28% of respondents). These findings demonstrate that respondents hit sticking points in project-level tasks prior to development.

Within the development process itself, there are still areas identified for improvement, however. Twenty percent of respondents reported getting stuck during debugging/ troubleshooting or testing, and testing was also one of the most-selected areas where respondents want better tools (28%).

MOST CHALLENGING STAGES



IMPROVEMENT IN TOOLS

28%

of respondents desired better tools for the planning stage

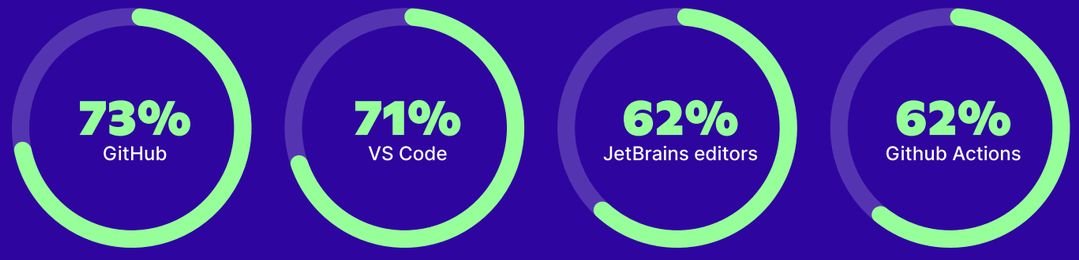
20%

reported getting stuck during debugging/ troubleshooting or testing

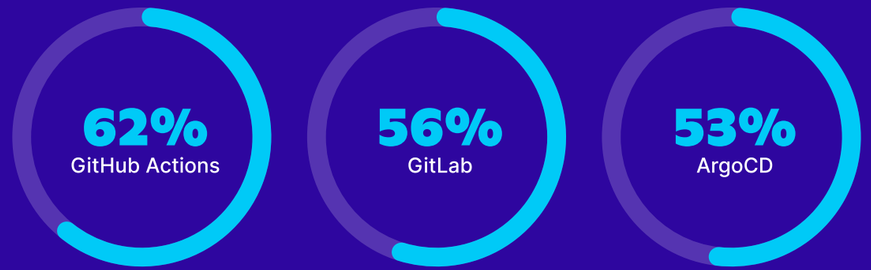
FAVORITE TOOLS OF THE TRADE

When we asked respondents to rate the tools they use, the highest percentage of positive responses (the proportion of good/very good ratings out of total responses) were as follows:

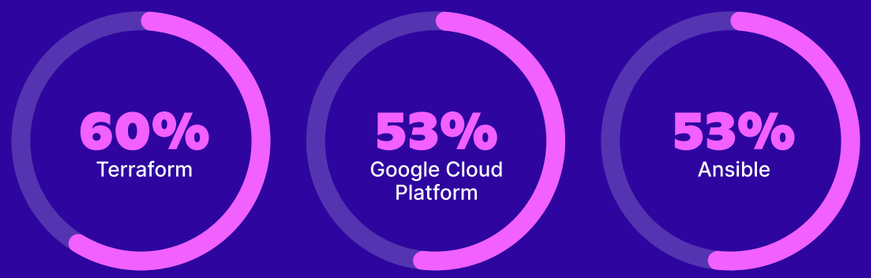
DEVELOPMENT TOOLS



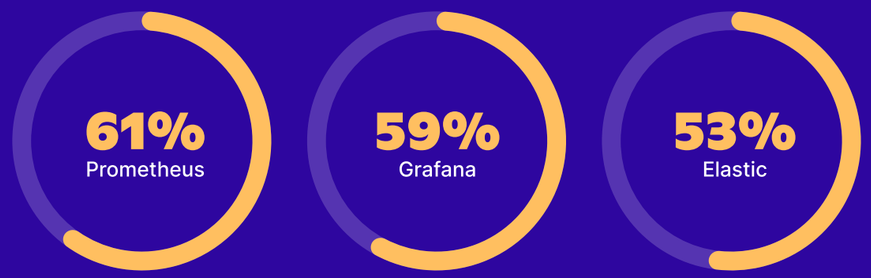
CI/CD TOOLS



PROVISIONING TOOLS



MONITORING TOOLS



THE DEVELOPMENT PROCESS

On the process and operations side, we asked respondents to rate their experiences along a number of dimensions, including their company/organization's development processes and operational efficiency.

The three development processes that garnered the most positive responses were ease of writing/editing code once the environment is ready (55%), continuous integration and debugging in development (tied at 53%), and ease of deploying changes (51%).

The processes that earned the most negative responses were code base documentation and maintenance (32%) and debugging in production (29%). It is interesting to note the stark contrast in perceptions of debugging in development (over half of respondents viewed this process positively) versus debugging in production (only about one-third viewed this process positively). This highlights the unique challenges and complexity associated with debugging in production environments.

KEY TAKEAWAYS

AREAS MOST POSITIVELY RATED

55%
Ease of writing/editing code once the environment is ready

53%
Continuous integration

53%
Debugging in development

51%
Ease of deploying changes

VS

AREAS MOST NEGATIVELY RATED

32%
Code base documentation and maintenance

29%
Debugging in production

EVALUATING DEVELOPMENT OPERATIONS

When we asked respondents to rate their company or organization's operational efficiency, the most positively rated options were feeling empowered to experiment and take risks (49%), collaboration among team members and other stakeholders (48%), and learning and improving as a team (for example, through retrospectives and post mortems) (46%).

Areas of least efficiency that were top-selected were balance of tech debt vs. feature work (32%) and uninterrupted time for deep work (28%).

This pattern of responses demonstrates interpersonal team-level factors are generally viewed positively; the inefficiencies respondents see tend to stem from the content and structure of their work.

RATING OPERATIONAL EFFICIENCY

AREAS MOST POSITIVELY RATED

49%
Feeling empowered to experiment and take risks

48%
Collaboration among team members and other stakeholders

46%
Learning and improving as a team
(for example, through retrospectives and post mortems)

VS

AREAS OF LEAST EFFICIENCY

32%
Balance of tech debt vs. feature work

28%
Uninterrupted time for deep work

AI'S EXPANDING ROLE IN APPLICATION DEVELOPMENT

AI ON THE RISE

The uptake of AI tools such as ChatGPT, GitHub Copilot, and Gemini (formerly Bard) among developers underscores AI's value in the development process. The survey sheds light on how developers are adopting and using AI and reflects a shift toward more intelligent, efficient, and adaptable development methodologies. The transformation is part of a larger trend observed across the tech industry as AI becomes increasingly central to software development.

KEY TAKEAWAYS

AI SENTIMENTS

40%

identify GenAI as an important trend in software development

38%

view AI assistants for software engineering as a key industry trend

IDENTIFYING THE HOTTEST AI INNOVATIONS

When we asked respondents what they view as the most important trends in the industry, Generative AI (40%) and AI assistants for software engineering (38%) emerged as the top-selected options.

Interestingly, more senior developers (back-end, front-end, and full-stack developers with over five years of experience) tended to view GenAI as most important, whereas more junior developers (less than five years' experience) identified AI assistants for software engineering as most important. This difference may signal varied and unique uses of AI throughout a career in software development.

WHAT RESPONDENTS VIEWED AS THE MOST IMPORTANT TRENDS IN THE INDUSTRY

40%

Generative AI

38%

AI assistants for software engineering

FEELINGS AROUND AI

AI is clearly trendy, but how does the industry really feel about it? Most respondents reported positive perceptions, with 65% saying they feel AI is a positive option, 61% agreeing it makes their jobs easier, and 55% saying it allows them to focus on more important tasks. A much smaller number of respondents view AI as a threat to their jobs (23%) or say it makes their jobs more difficult (19%).

KEY TAKEAWAYS

HOW DO RESPONDENTS FEEL ABOUT AI?

65% agree that AI is a positive option

61% agree that AI makes their jobs easier

55% agree that AI allows them to focus on more important tasks

VS

23% see AI as a threat to their jobs

19% say it makes their jobs more difficult

AI: OVERRATED OR UNDERRATED?

Curiously, despite high usage and generally positive feelings towards AI, 45% of respondents also reported they feel AI is over-hyped. This may dovetail with the finding that relatively few respondents subscribe to the (arguably hyped) view that AI will replace them anytime soon — despite it being a key tool for their work.

45%

of respondents reported they feel AI is over-hyped

AI AND APP DEVELOPMENT

We asked respondents what they use AI for, how dependent they feel on AI, and what AI tools they use most often. Most respondents (64%) report already using AI for work, underscoring AI's penetration into the software development field. More specifically, they most often use AI to write code (33%), write documentation (29%), do research (28%), write tests (23%), troubleshoot/debug (21%), and for CLI commands (20%).

- **33%** for writing code
- **23%** for writing tests
- **29%** for writing documentation
- **21%** for troubleshooting/debugging
- **28%** for research
- **20%** for CLI commands

This trend is notably driven by junior/mid-level developers and DevOps/Platform Engineers, who express a higher dependency on AI compared to senior developers surveyed.

KEY TAKEAWAYS

64%

of respondents report using AI for work

46% report using ChatGPT

30% report using GitHub Copilot

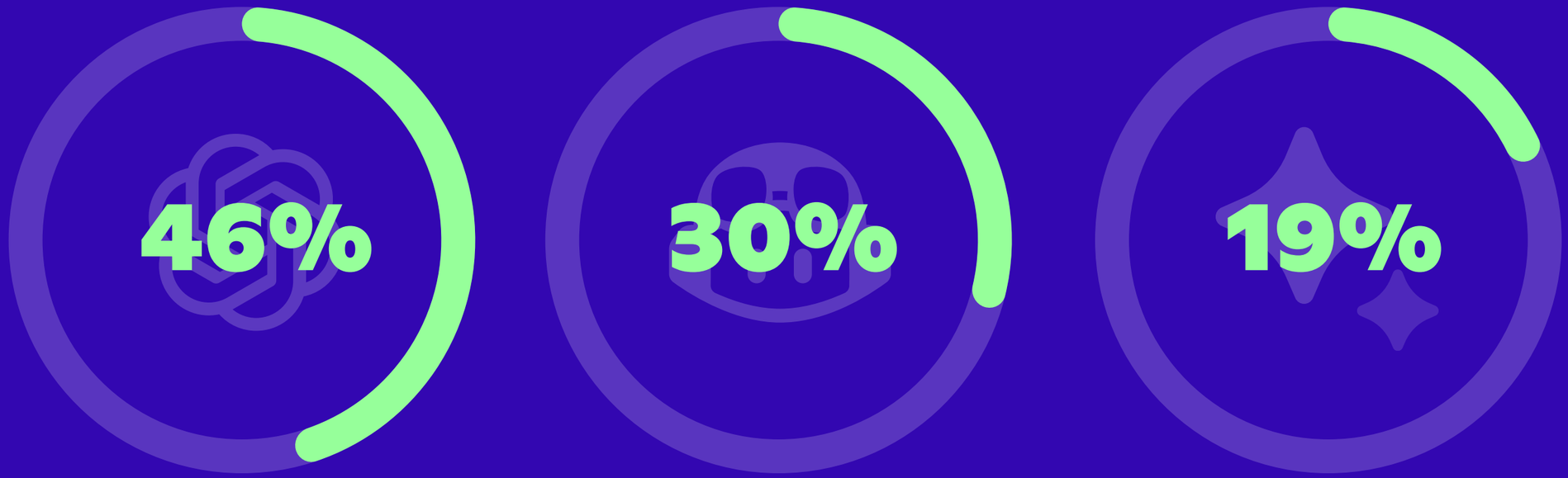
19% report using Gemini (formerly Bard)

RATING RELIANCE ON ARTIFICIAL INTELLIGENCE

For the 568 respondents who indicated they use AI for work, we also asked how dependent they feel on AI to get their job done on a scale of 0 (not at all dependent) to 10 (completely dependent). Responses ranged substantially and varied by role and years of experience, but the overall average reported dependence was about 4 out of 10, indicating relatively low dependence.

IDENTIFYING THE MOST POPULAR AI TOOLS

In terms of AI tools, respondents most often use ChatGPT (46%), GitHub Copilot (30%), and Gemini (formerly Bard) (19%).



ChatGPT

GitHub Copilot

Gemini
(formerly Bard)

KEY TAKEAWAYS

GROWING INTEREST IN MACHINE LEARNING

This year's survey showed a growing interest in ML engineering and data science within the Docker community, signaling a broader acceptance and integration of AI/ML technologies within software development.

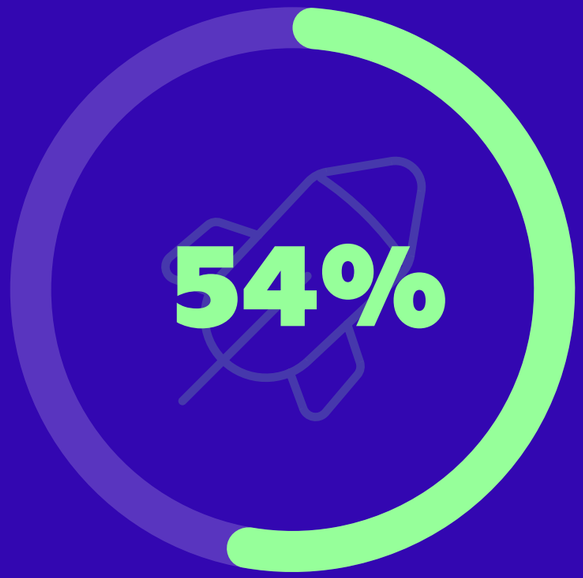
46%

are working on ML in some capacity

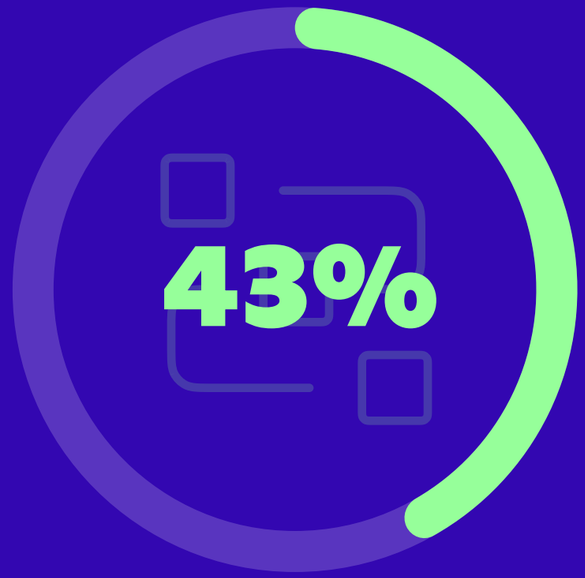
- **54%** train and deploy ML models in one or more projects
- **43%** work on ML infrastructure
- **39%** leverage pre-trained ML models

MACHINE LEARNING USAGE IN DEVELOPMENT

When asked if they are working on ML in any capacity, almost half of respondents (46%) replied in the affirmative. Within that group, 54% said that they trained and deployed ML models in one or more projects, 43% worked on ML infrastructure, and 39% leveraged pre-trained ML models.



trained and deployed ML models in one or more projects



worked on ML infrastructure

leveraged pre-trained ML models

SECURITY IN APPLICATION DEVELOPMENT

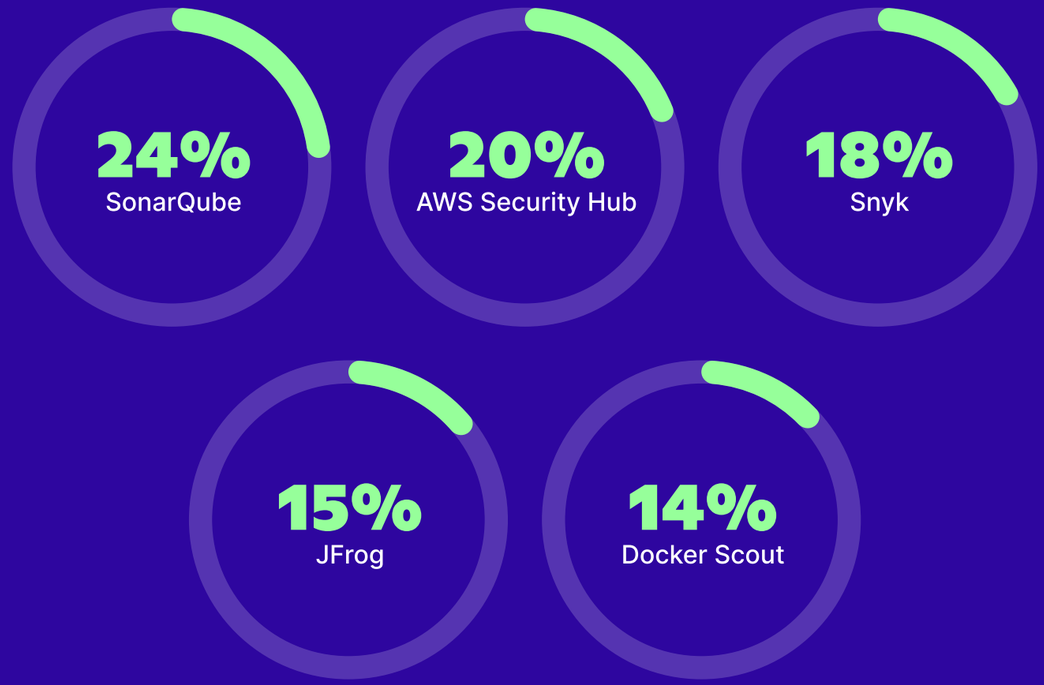
SECURITY TOOLS AND TASKS

With cybercrime rates continuing to soar, software development projects are vulnerable to cyberattacks, especially in the development, launch, and maintenance stages. In the quest to ship better software, the resulting shift-left approach to security poses both challenges and opportunities for developers. We asked respondents which security tools they use, what security-related tasks they deal with, and what roles/teams focus on software security at their company or organization.

Asked which security tools they use, respondents most often cited SonarQube (24%), followed by AWS Security Hub (20%), Snyk (18%), JFrog (15%), and Docker Scout (14%).

KEY TAKEAWAYS

MOST USED SECURITY TOOLS



MOST COMMON SECURITY TASKS

COMMON SECURITY TASKS

AMONG THE SECURITY TASKS CARRIED OUT BY RESPONDENTS, FIXING VULNERABILITIES WAS BY FAR THE MOST COMMON

In fact, fixing vulnerabilities is clearly a key responsibility that is shouldered by many different individuals within an organization; it was the top security task reported by back-end, front-end, and full-stack developers, DevOps and Platform engineers, and most security-focused roles (e.g., Security Manager, DevSecOps).

Other common tasks included running security scans (34%) and dealing with the scan results (32%), ahead of monitoring security incidents (30%) and logging data analysis (26%).

49%

Fixing vulnerabilities

- **34%** Running security scans
- **32%** Dealing with the scan results
- **30%** Monitoring security incidents
- **26%** Logging data analysis

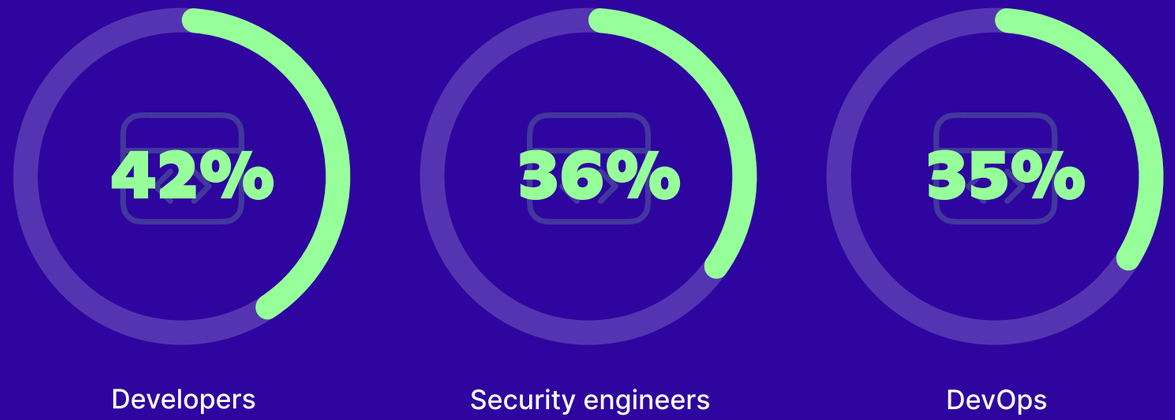
KEY TAKEAWAYS

SHIFTING SECURITY LEFT

Asked what roles/teams focus on software security, developers top the list (42%), followed by security engineers (36%), DevOps (35%), DevSecOps (28%), and platform engineers (25%).

The finding that developers handle security provides evidence of security's shift to the left. Yet only 14% of respondents viewed shifting security to the left as an important industry trend. Why might this be? Check out the next slide for a possible explanation.

ROLES/TEAMS FOCUSED ON SOFTWARE SECURITY



34% OF RESPONSES RATED SECURITY-RELATED TASKS AS:

difficult **very difficult**

somewhat difficult moderate

somewhat easy easy very easy

When we asked respondents to rate tasks at their company or organization, security-related tasks topped the list of those deemed difficult/very difficult, with 34% of respondents selecting one of these options. And when asked where better tools were needed in the development process, security/vulnerability remediation tools were the fourth-most selected, with 25% of those surveyed responding affirmatively.

While it's not entirely clear what larger trends these findings point to, it seems likely that the shift-left approach to security is a source of frustration for developers and an area where more effective tools could make a difference.

TOOL IMPROVEMENTS NEEDED

#1 Testing

#2 Planning

#3 Monitoring/Logging/Maintenance

#4 Security/Vulnerability Remediation

#5 Continuous Delivery/Deployment

#6 Estimation

#7 Debugging/troubleshooting

#8 Designing

#9 Continuous Integration (CI)

#10 Writing code

#11 PR review

#12 Provisioning

#13 Building/Compiling

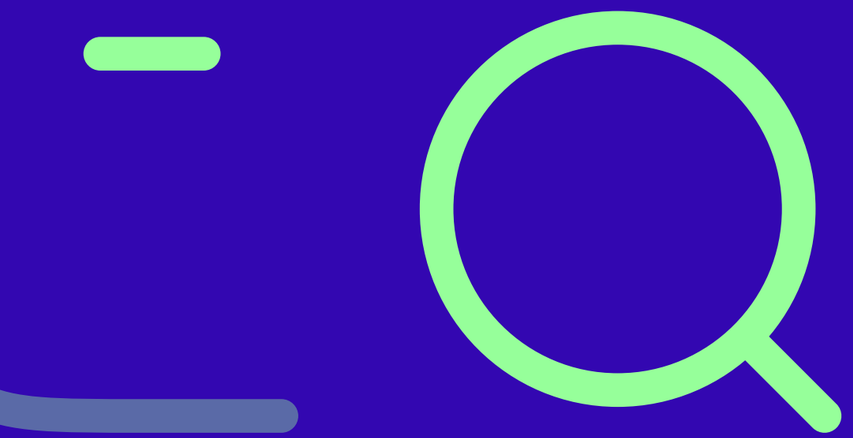
CONCLUSION

CONCLUSION

Operating on the cusp of churning technological change, application developers are beset with dynamic needs. Docker's approach to addressing these needs reflects its commitment to continuous improvement and innovation. The 2024 Docker State of Application Development Report exposes a rich vein of data-driven insights that Docker is poised to mine to enhance its offerings, with a view to helping developers, teams, and organizations thrive in an increasingly complex and ever-changing technology landscape.

In spotlighting key trends such as the expanding roles of the cloud and AI/ML in software development, the report not only freeze-frames the current state of application development but also charts a course for Docker's continued evolution. Looking ahead, Docker will continue to monitor these trends and adapt its offerings accordingly. As the industry evolves, so will Docker's role in driving the future of application development.

METHODOLOGY



METHODOLOGY

The 2024 Docker State of Application Development Report was an online, 20-minute survey conducted by Docker's User Research Team in the fall of 2023. This was the second annual Docker State of Application Development survey, and the third will take place in the fall of 2024.

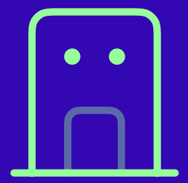
The survey was developed to inform Docker's product strategy. Given the fascinating information we discovered, we wanted to share the findings with the community.

Questions ranged from demographic information, how respondents like to learn and discover new things, the tools they use, and their perspectives on trends, tools, and work processes.

FROM HOBBYISTS TO ENTERPRISE PROS:
WHO TOOK THE SURVEY?

42%

work for a small company
(up to 100 employees)



28%

work for a mid-sized company
(between 100 and 1000 employees)

25%

work for a large company
(more than 1000 employees)

Survey respondents ranged from home hobbyists to professionals at companies with more than 5,000 employees. Forty-two percent of respondents work for small companies (up to 100 employees), 28% work for mid-sized companies (between 100 and 1,000 employees), and 25% work for large companies (more than 1,000 employees).

Over half of respondents had engineering roles, with 36% identifying as back-end or full-stack developers; 21% as DevOps, infrastructure managers, or platform engineers; and 4% as front-end developers. Other roles included dev/engineering managers, company leadership, product managers, security roles, and AI/ML roles. There was nearly an even split between respondents with more experience (6+ years, 54%) and less experienced (0-5 years, 46%).

A CHANGING USER BASE

Our report underscored a marked growth in roles focused on machine learning (ML), engineering, and data science within the Docker ecosystem. These roles made up 8% of respondents in the latest survey, up from about 1% in our 2022 survey. ML engineers and data scientists represent a rapidly expanding user base, which signals the growing relevance of AI to the software development field, and the blurring of the lines between tools used by developers and tools used by AI/ML scientists.

More than 34% of respondents said they work in the computing or IT/SaaS industry, but we also saw responses from individuals working in accounting, banking, or finance (8%); business, consultancy, or management (7%); engineering or manufacturing (6%); and education (5%). Other responses came in from professionals in a wide range of fields, including media, academic research, transport or logistics, retail, charity or volunteer work, healthcare, construction, creative arts or design, environment or agriculture, and marketing, advertising, or PR.

CREDITS

This research was designed, conducted, and analyzed by the Docker UX Research Team: Rebecca Floyd, Ph.D.; Julia Wilson, Ph.D.; and Olga Diachkova.

For the complete details on methodology, contact uxresearch@docker.com

SUBSCRIBE TO OUR NEWSLETTER!

To help you stay up to date with the ever-evolving world of developer tools and container technology, subscribe to the Docker Newsletter.