ISSN: 2473-4901
v10 n6119

Navigating the AI Landscape: An Analysis of AI Developer Tools Usage, Sentiment, and Trust Among IT Professionals

Wendy Ceccucci
wendy.ceccucci@qu.edu
Business Analytics and Information Systems Department
Quinnipiac University
Hamden, CT 06518 USA

Alan Peslak
arp14@psu.edu
Department of Information Sciences and Technology
Penn State University
Dunmore, PA 18512 USA

Kiku Jones
kiku.jones@qu.edu
Business Analytics and Information Systems Department
Quinnipiac University
Hamden, CT 06518 USA

Abstract

GitHub Copilot, developed by GitHub, is a new tool aiding developers with a range of tasks, including the generation of code snippets, documentation assistance, and the formulation of implementation strategies. Comparable AI development tools, such as Tabnine and AWS Code Whisperer, also serve as developmental aids but are utilized to varying and much lesser extents. Our research examines the adoption of GitHub Copilot and similar AI development tools among professional developers and other users using the Stack Overflow Annual Survey. The study reveals notable age-related disparities in tool usage, with younger individuals showing a markedly higher propensity to employ these technologies compared to older users. Additional significant insights include variations in usage based on the type of developer, professional status, and the influence of user attitudes towards AI on the adoption of GitHub Copilot among developers.

Keywords: AI Developer Tools, Artificial Intelligence, GitHub Copilot, Technology Adoption, AI Trust

1. INTRODUCTION

The integration of Artificial Intelligence (AI) in software development has marked a transformative era in the information technology (IT) sector. AI developer tools, ranging from automated code assistants to advanced debugging algorithms, have become pivotal in shaping how IT professionals approach software development. This paper presents a comprehensive analysis of the usage patterns, sentiment, and trust levels associated with AI

developer tools among IT developers. The study aims to provide an understanding of how these tools are reshaping the development landscape, the attitudes of developers towards them, and the trust issues that may arise.

The advent of AI in development tools has introduced both opportunities and challenges. While these tools promise increased efficiency, accuracy, and even creativity in coding, they also raise questions about reliability, ethical use, and the potential for diminishing human skill sets. Understanding the sentiments and trust levels of developers towards AI tools is crucial for the future design, implementation, and governance of these technologies. According to a developer survey by GitHub (2024) 92% of developers working in large companies are already using AI tools either at work or in their personal time.

This paper begins by exploring the current landscape of AI developer tools, categorizing them based on their functionalities and the aspects of software development they impact. We then delve into the methodologies used to gauge the usage patterns of these tools among IT professionals. This includes a survey of a diverse group of developers, encompassing various industries, experience levels, and geographical locations.

Following the usage analysis, the paper addresses the core aspects of sentiment and trust. Sentiment analysis is conducted through survey responses. This analysis provides insights into the general attitudes of developers towards these tools, ranging from enthusiasm and optimism to skepticism and concern. Trust, a more complex and multifaceted issue, is examined through a separate survey response.

The findings of this study are expected to offer valuable insights for multiple stakeholders. Tool developers and AI researchers can gain a better understanding of user attitudes and trust factors, guiding them in creating more user-centric and trustworthy tools. Organizations and team leaders can use these insights to make informed decisions about integrating AI tools into their workflows, considering both the technical and human aspects. Additionally, the study contributes to the broader discourse on the role of AI in the future of work, particularly in the IT sector.

This paper aims to shed light on the complex dynamics between IT developers and AI developer tools, focusing on usage patterns, sentiments, and trust. By providing an analysis of these aspects, the study seeks to contribute to the responsible and effective integration of AI in software development practices.

2. LITERATURE REVIEW

AI Code Generation Tools

Ai code generation systems are simply AI systems trained to write code, thereby automating the process of programming. The code generating system is built upon machine learning algorithms that train on large datasets of code. The algorithms learn the common patterns, practices, and conventions in coding and then can generate new code based on their learning.

AI coding tools are becoming standard practice for many software developers. According to an online survey conducted by Wakefield Research on the behalf of GitHub, 92% of U.S. developers are using AI coding tools both in and outside of work (Shani & GitHub Staff, 2023). The survey responses were from 500 U.S. based developers who were not managers and worked at companies with 1,000-plus employees. They also found that more than 4 out of 5 developers expected AI coding tools would make their team more collaborative.

The most widely used AI developer tool is GitHub's Copilot which was released for technical preview in 2021. GitHub Copilot is an AI-powered code assistant developed by GitHub in collaboration with OpenAI. GitHub Copilot is designed to help developers write better code faster and with fewer errors. It operates by analyzing the context within the codebase and suggesting whole lines or blocks of code that developers can incorporate or modify as needed.

GitHub Copilot, an AI-powered code assistant, is used by software developers primarily in JavaScript and Python, with Visual Studio Code being the main IDE. It is most commonly used for data processing and code generation, with the potential to significantly reduce development time and slightly increase code quality. However, concerns about security and difficulty of integration have been raised, and developers are divided in their opinions about the tool (Jaworski & Piotrkowski, 2023; Zhang, et al., 2023).

Other examples of AI based code assistants include Tabnine (Kedar, 2024), and CodeWhisperer (Desai & Deo 2022). Comparing the tools:

1. GitHub Copilot is owned by Microsoft and is based on OpenAI's Codex model. One of its strengths is its flexibility in

ISSN: 2473-4901
v10 n6119

deployment (cloud vs local). It excels in its deep integration with GitHub and its powerful language model but is best suited for those already invested in the GitHub ecosystem.
2. Tabnine based on various models including GPt-3. It also offers flexibility in deployment and a wide range of IDE and language support, which is beneficial for diverse development environments.
3. Amazon CodeWhisperer is developed by Amazon Web Services (AWS). It is based on AWS's proprietary models, including adaptations of large language models. The context-aware code suggestions are similar to the other tools but tailored for cloud and serverless environments.

According to a Survey by CodeSignal (2024) of 1,000 developers worldwide, 81% of developers surveyed said they use AI-powered coding assistants. The number one reason for using the tool was for learning new technical skills or knowledge. The second most common reason was for generating boilerplate code.

Some of the advantages of AI code generation tools include:

1. Saves time. These tools can accelerate the development cycle, making the process more efficient.
2. Learning and skill development. AI code generation supports learning and skill enhancement. By observing the AI-generated code developers can learn new techniques, understand different code structures, and adopt better coding practices.
3. Accessibility for non-experts. Individuals with less coding knowledge can create functional applications by simply describing the requirements in natural language.
4. Reduces human error. AI tools can help minimize human errors and the AI-generated code is often well-documented with explanations making it easier to understand and debug.

Along with its potential advantages, AI code generated tools have some limitations. These include:

1. Quality and accuracy. AI-generated code may not always meet the quality and accuracy standards required for complex projects (Hasselbring & Reussner, 2006; Lipner, 2004).
2. Contextual understanding. AI tools often fail to fully grasp specific project contexts, resulting in misaligned code (Wang, et al., 2024).
3. Dependency on data. The effectiveness of AI code generation tools is highly dependent on the quality and breadth of their training data. Poor or biased data leads to poor performance.
4. Security concerns. AI-generated code may introduce vulnerabilities if secure coding practices are not properly embedded. According to a study by Snyk (2023), more than three-quarters of developers bypass established protocols to use code completion tools despite potential risks.
5. Limited creativity. AI lacks the creativity and problem-solving abilities inherent to human developers, leading to less innovative solutions.
6. Trust and reliability. Developers often find it challenging to trust AI-generated code without extensive testing and verification, increasing their workload (Wang, et al., 2024).
7. Ethical and legal issues. The use of AI in code generation raises concerns related to intellectual property and the legality of using certain code snippets.

According to a study by Dakhel, et al. (2023), Copilot can be an asset in software projects when used by expert developers, as its suggestions can match human contributions in quality. However, Copilot can become a liability if used by novice developers who may struggle to filter its suggestions effectively.

Trust and AI Tools

Recent studies have explored the impact of AI-powered code generation tools on software developers' trust and the potential for these tools to automate routine programming tasks (Cheng et al., 2024; Ernst & Bavota, 2022). These tools, such as GitHub Copilot, have the potential to increase the level of abstraction in software development, allowing developers to focus on business processes rather than code (Palacios-González, et al., 2008). However, concerns have been raised about the potential for bias, legal compliance, and security vulnerabilities in AI-driven development environments (Ernst & Bavota, 2022).

One significant challenge involves helping users evaluate the trustworthiness of AI tools. The importance of software developers' trust in

programming has been extensively studied, highlighting it as a crucial design requirement for these tools. Trust is considered a key prerequisite for ensuring the safety of the resulting software products (Hasselbring & Reussner, 2006; Lipner, 2004).

According to a study by Wang, et al. (2024), developers' trust is rooted in the AI tool's perceived ability, integrity, and benevolence, and is situational, varying according to the context of usage. Existing AI code generation tools lack the affordances for developers to efficiently and effectively evaluate the trustworthiness of AI-powered code generation tools.

Several other demographics and factors can influence a developer's trust in AI tools, and understanding these can help tailor AI solutions to better meet user needs. These factors include developer's geographical location, gender, and socioeconomic status.

Developers from different regions may have varying levels of trust in AI tools based on cultural attitudes towards technology and data privacy. Although their research did not focus on developers specifically, Grassini and Ree (2023) found that respondents from the USA demonstrated higher levels of hopefulness for AI technology compared to those from the UK. Their finding suggests that cultural context does play a role in shaping individuals' perceptions of AI technology.

In the same study, they found that male respondents showed higher hopes for AI systems than female respondents. This finding is consistent with previous research, which reported that males perceive AI as more useful (Arujo, et al., 2020) and generally more favorable than females (Lozano, et al., 2021). Likewise, a study by Armutat, et al. (2024) found that men tended to view AI applications more positively, rate their own AI competencies higher, and have more trust in the technology compared to women.

Age and AI Coding Tools

The integration of AI tools into the workflow of software developers has shown a notable age-related trend, with younger developers demonstrating a higher propensity to adopt such technologies. This trend can be attributed to several factors. First, younger developers are generally more exposed to the latest technological innovations during their education and early careers, making them more receptive to AI-driven solutions. Educational institutions are increasingly incorporating AI and machine learning courses into their curricula, which equips new graduates with the skills and familiarity needed to leverage these tools effectively.

Moreover, the rapid pace of technology means that younger individuals often have a 'digital native' advantage. They tend to be more adaptive to changes in the tech landscape, including the use of sophisticated AI platforms that require a steep learning curve. A report by GitHub, The State of the Octoverse (Daigle & GitHub Staff, 2024), highlights that newer programmers are more likely to utilize AI coding assistants, attributing this to their up-to-date training and inherent flexibility in adopting new workflows.

Additionally, the software development industry itself fosters a culture of innovation and continuous learning, which resonates well with the mindset of younger developers. They are often driven by the potential to streamline coding processes, enhance productivity, and solve complex problems efficiently with the help of AI tools. In practical terms, younger developers utilize AI tools for a range of tasks including writing and reviewing code, automating repetitive tasks, and even in conceptual phases of development like brainstorming and prototyping. The convenience and efficiency offered by AI tools align with the fast-paced, results-oriented environment preferred by this demographic. The influence of AI on younger developers is not just transforming their individual workflows but is also shaping the future dynamics of team collaborations and project management in software development. As these young developers progress in their careers, their preference for AI-enhanced workflows is likely to catalyze broader adoption of these technologies across the industry, potentially leading to more innovative solutions in the overall approach to software development.

3. METHODOLOGY

This study explores the usage of AI tools by software developers by leveraging data from the 2023 Stack Overflow survey. The Stack Overflow Developer Survey is widely acknowledged as the most comprehensive survey targeting coders worldwide. Annually, it covers a wide range of subjects, from developers' technology preferences to their career aspirations. According to Stack Overflow's website:

> "For 13 years, we've delivered industry-leading insights regarding the developer community. This is the voice of the developer. Analysts, IT leaders, reporters, and other developers turn to

ISSN: 2473-4901
v10 n6119

> this report to stay up to date with the evolving developer experience, technologies that are rising or falling in favor, and to understand where tech might be going next." (Stack Overflow, 2023)

The validity of using Stack Overflow data is supported by its frequent citation in numerous peer-reviewed publications, including studies by Barua, et al. (2014), Asaduzzaman, et al. (2013), and Treude and Robillard (2016). The dataset comprises a rich mix of demographic data, descriptive statistics, and responses to opinion-based questions about the programming industry. IBM SPSS 29 was utilized for the data analysis. Primarily descriptive statistics and crosstab analyses were used in the study.

4. RESULTS

Survey respondents were asked, "Which AI-powered developer tools did you use regularly over the past year and which do you plan to work with over the next year?"

Table 1 displays the results of this survey question for all participants. Out of 89,870 total responses, 686 respondents did not answer the question. This leaves a total of 89,184 valid responses.

AI Developer Tool	Use		Do Not Use	
GitHub Copilot	22,078	24.76%	67,106	75.24%
Tabnine	5,193	5.82%	83,991	94.18%
AWS CodeWhisperer	2,071	2.32%	87,113	97.68%
Synk Code	538	0.60%	88,646	99.40%
Codeium	504	0.57%	88,680	99.43%
Whispr AI	455	0.51%	88,729	99.49%
Rubber Duck.AI	151	0.17%	89,033	99.83%
Mintify	-	0.00%	89,184	100.00%
Replit Ghostwrite	-	0.00%	89,184	100.00%

Table 1: AI- Powered Developer Tool Usage by All Respondents

In the next table, Table 2, the responses were limited to software developers only. A total of 67,973 respondents identified as software developers by profession. With 686 respondents not answering the question, the total number of valid responses was 67,287.

AI Developer Tool	Use		Do Not Use	
GitHub Copilot	17,432	25.93%	49,805	74.07%
Tabnine	3,651	5.43%	63,586	94.57%
AWS CodeWhisperer	1,527	2.27%	65,710	97.73%
Synk Code	413	0.61%	66,824	99.39%
Codeium	333	0.50%	66,904	99.50%
Whispr AI	280	0.42%	66,957	99.58%
Rubber Duck.AI	79	0.12%	67,158	99.88%
Mintify	-	0.00%	67,237	100.00%
Replit Ghostwrite	-	0.00%	67,237	100.00%

Table 2: AI- Powered Developer Tool Usage by Software Developers

The main AI developer tool is clearly GitHub Copilot, with approximately 25% usage by both groups. The next most used tool is Tabnine, at about 5%. Given the dominance of GitHub Copilot, the focus of the rest of our paper will be on this tool.

The next question related to AI development tools usage asked the question: "Do you currently use AI tools in your development process?" The three possible responses were "Yes", "No, but I plan to soon", and "No, I don't plan to". Table 3 shows the results. Overall, the study reveals that 44% of Developers currently use AI tools and another 26% plan to soon. Only 30% do not have plans to use AI.

Response	Count	Percent
Yes,	29,697	44.20%
No, but I plan to	17,401	25.90%
No, I don't plan to	20,139	30%
Total	67,237	100%

Table 3: Plans to Use AI- Powered Developer Tool by Software Developers

The survey then looked at how respondents viewed AI (Table 4). Overall, 76% of developers have a favorable view of AI. Table 5 shows the effect of attitude towards AI and the use of GitHub Copilot. Those with a very favorable view use Copilot nearly 50%. This rate drops for other views hovering in the 20% range for those with neutral or unfavorable views.

ISSN: 2473-4901
v10 n6119

Attitude	Frequency	Percent	Valid Percent
Very Favorable	13,002	19.30%	27.70%
Favorable	22,717	33.80%	48.40%
Indifferent	9,705	14.40%	20.70%
Unfavorable	1,305	1.90%	2.80%
Very Unfavorable	199	0.30%	0.40%
Total	46,928	69.80%	100%
Missing	20,309	30.20%	
Total	67,237	100%	

Table 4: Software Developer's Attitudes Toward AI

		GitHub Copilot		
Attitude	Statistic	Use	Do Not Use	Total
Very Favorable	Count	6,421	6,581	13,002
	Percentage	49.4%	50.6%	
Favorable	Count	7,768	14,949	22,717
	Percentage	34.2%	65.8%	
Indifferent	Count	1,935	7,770	9,705
	Percentage	19.9%	80.1%	
Unfavorable	Count	292	1,013	1,305
	Percentage	22.4%	77.6%	
Very Unfavorable	Count	54	145	199
	Percentage	27.1%	72.9%	
Total	Count	16,470	30,458	46,928
	Percentage	35.1%	64.9%	

Table 5: Software Developer's Attitudes Toward AI and use of GitHub Copilot

The survey then asked respondents "How much do you trust the accuracy of the output from AI tools as part of your development workflow?" Table 6 shows the survey results. Sentiment is higher than trust in the accuracy of AI output with only 2% of developers having high trust. There is a large percentage who somewhat trust AI at 38%. Twenty-eight percent distrust AI and 32% are neutral.

Level of Trust	Frequency	Percent	Valid Percent
Highly Trust	1,139	1.7%	2.4%
Somewhat Trust	17,696	26.3%	37.8%
Neither Trust Nor Distrust	14,789	22.0%	31.6%
Somewhat Distrust	10,514	15.6%	22.4%
Highly Distrust	2,721	4.0%	5.8%
Total	46,859	69.7%	100.0%
Missing	20,378	30.3%	
Total	67,237	100.0%	

Table 6: Software Developer's Attitudes Toward AI and use of GitHub Copilot

The effect of trust on the usage of GitHub Copilot is revealing. Lack of trust negatively affects adoption with 39% of those who at least somewhat trust AI output using GitHub Copilot. But the percentage of developers with neutral or lack of trust only drops to about 32%. This finding, though significant, demonstrates that trust has a limited impact on the use of AI development tools.

		GitHub Copilot		
Level of Trust	Statistic	Use	Do Not Use	Total
Highly Trust	Count	448	691	1,139
	Percentage	39.3%	60.7%	
Somewhat Trust	Count	6,957	10,739	17,696
	Percentage	39.3%	60.70%	
Neither Trust Nor Distrust	Count	4,898	9,891	14,789
	Percentage	33.10%	66.90%	
Somewhat Distrust	Count	3,286	7,228	10,514
	Percentage	31.30%	68.70%	
Highly Distrust	Count	874	1,847	2,721
	Percentage	32.10%	67.90%	
Total	Count	16,463	30,396	46,859
	Percentage	35.10%	64.90%	

Table 7: Software Developer's Trust in AI and use of GitHub Copilot

Age is clearly a factor in the use of GitHub Copilot (Table 8). This result has been supported by the literature. Nearly one half of the developers under 18 use it and the percentage declines for each older age group. The 18-24 users check in at 35%, 25-34 at 27% and so on. The oldest group over 65 only has a 6% participation rate.

		GitHub Copilot		
Age	Statistic	Use	Do Not Use	Total
Under 18 years old	Count	226	196	422
	Percentage	53.6%	46.4%	
18-24 years old	Count	3,822	7,180	11,002
	Percentage	34.7%	65.3%	
25-34 years old	Count	7,692	21,156	28,848
	Percentage	26.7%	73.3%	
35-44 years old	Count	4,172	13,132	17,304
	Percentage	24.1%	75.9%	
45-54 years old	Count	1,217	5,270	6,487
	Percentage	18.8%	81.2%	
55-64 years old	Count	277	2172	2449
	Percentage	11.3%	88.7%	
65 years or older	Count	38	556	594
	Percentage	6.4%	93.6%	
Prefer Not to Say	Count	18	113	131
	Percentage	13.7%	86.3%	
Total	Count	49,805	17,432	67,237
	Percentage	74.1%	25.9%	

Table 8: Software Developer's Age and use of GitHub Copilot

When we examine all respondents, we find that both hobbyists and developers by profession have the highest use of GitHub Copilot (Table 9).

ISSN: 2473-4901
v10 n6119

Surprisingly, those learning to code are lower than these groups at 23%.

		GitHub Copilot		
Developer Level	**Statistic**	**Use**	**Do Not Use**	**Total**
I am a developer by profession	Count	17,432	49,805	67,237
	Percentage	25.9%	74.1%	
I am learning to code	Count	1,161	3,800	4,961
	Percentage	23.4%	76.6%	
I am not primarily a developer, but I write code sometimes as part of my work/studies	Count	1,835	7,119	8,954
	Percentage	20.5%	79.5%	
I code primarily as a hobby	Count	1,318	3,642	4,960
	Percentage	26.6%	73.4%	
I used to be a developer by profession but no longer am	Count	332	1,529	1,861
	Percentage	17.8%	82.2%	
None of these	Count	0	1211	1211
	Percentage	0.0%	100.0%	
Total	Count	22,078	67,106	89,184
	Percentage	24.8%	75.2%	

Table 9: Developer's Level and use of GitHub Copilot

When we examine the developer type and use of GitHub Copilot, we find that Blockchain, Developer Advocates, and Front-end developers have the highest usage rate. They all exceed 30% usage (Appendix 1). Database, enterprise, and embedded developers all are lowest end at less than 15%.

5. CONCLUSIONS

GitHub Copilot was found to be used the most of all AI powered developer tools regardless of a person's profession. When looking strictly at software developers, the same result was found with GitHub Copilot being used by 25% of the respondents and the next tool coming in at only 5%. An additional finding to help tool developers is that a majority of developers have a favorable view of AI and somewhat trust it. The favorable view of AI does seem to impact a developer's choice in using the AI developer tool. However, trust in AI appears to have limited impact. This is an area where future researchers will want to look further to determine what other factors are possibly moderating this relationship.

Age has a clear impact on the use of AI developer tools. As the age group increased, the use of AI developer tools decreased. This finding, which is supported in the literature, may be helpful to organizations as they review their workforce and determine potential training or motivating opportunities to engage different age groups to participate in using the AI developer tools. Another finding which can be helpful to organizations is that those learning to code do not use the AI developer tools as much as professional developers or even hobbyists. Showing the benefits of learning to code utilizing the AI developer tools and the time saving in completing projects will likely be a positive for both the organization and the developer. If there are developers on site who are already proficient in using the AI developer tools, it may be beneficial to the organization to have these experienced developers work with those who are just beginning to discover the AI developer tools. In addition, those developer types who have the highest usage rate of AI developer tools may be the best people to talk to the other developers who are not utilizing the tools to discuss their experiences to provide credibility to the recommendation to use the tools.

6. REFERENCES

Araujo, T., Helberger, N., Kruikemeier, S. & Vreese, C. (2020). In AI we trust? Perceptions about automated decision-making by artificial intelligence. AI & Society, 35, 611-623. https://doi.org/10.1007/s00146-019-00931-w

Armutat, S., Wattenber, M., & Mauritz, N. (2024). Artificial Intelligence – Gender-Specific Differences in Perception, Understanding, and Training Interest. *Proceedings of the 7th International Conference on Gender Research,* 7(1), 36-43. https://doi.org/10.34190/icgr.7.1.2163

Asaduzzaman, M., Mashiyat, A. S., Roy, C., & Schneider, K. (2013). Answering questions about unanswered questions of stack overflow. *Proceedings of the 10th Working Conference on Mining Software Repositories,* 97-100. https://doi.org/10.1109/MSR.2013.6624015

Barua, A., Thomas, S., & Hassan, A. (2014). What are developers talking about? An analysis of topics and trends in stack overflow. *Empirical Software Engineering*, *19*, 619-654. https://doi.org/10.1007/s10664-012-9231-y

Cheng, R., Wang, R., Zimmermann, T., & Ford, D. (2024). "It would work for me too": How Online Communities Shape Software Developers' Trust in AI-Powered Code Generation Tools. *ACM Transactions on Interactive Intelligent Systems*. 14(2), 1–39. https://doi.org/10.1145/3651990

CodeSignal (2024). *Trends Report Developers & AI Coding Assistant Trends*. CodeSignal.

ISSN: 2473-4901
v10 n6119

Retrieved July 26, 2024, from https://codesignal.com/report-developers-and-ai-coding-assistant-trends/

Daigle, K. & GitHub Staff (2024). *The State of the Octoverse.* GitHub. Retrieved on July 26, 2024, from https://GitHub.blog/news-insights/research/the-state-of-open-source-and-ai/

Desai, A. & Deo, A. (2022*). Introducing Amazon CodeWhisperer, the ML-powered coding companion*. AWS. Retrieved July 26, 2024, from https://aws.amazon.com/blogs/machine-learning/introducing-amazon-codewhisperer-the-ml-powered-coding-companion/

Dakhel, A., Majdinasab, V., Nikanjam, A., Khomh, F., Desmarais, M., & Jiang, Z. (2023). GitHub Copilot AI pair programmer: Asset or Liability?, *Journal of Systems and Software*, 203. https://doi.org/10.1016/j.jss.2023.111734

Ernst, N., & Bavota, G. (2022). AI-Driven Development Is Here: Should You Worry? *IEEE Software*, 39(2) 106-110. https://doi.org/10.48550/arXiv.2204.07560

GitHub (2024). *GitHub Developer Survey*. GitHub. Retrieved July 26, 2024, from https://GitHub.blog/news-insights/research/survey-reveals-ais-impact-on-the-developer-experience/#developers-want-more-opportunities-to-upskill-and-drive-impact

Grassini, S. & Ree, A. (2023). Hope or Doom AI-ttitude? Examining the Impact of Gender, Age, and Cultural Differences on the Envisioned Future Impact of Artificial Intelligence on Humankind. *Proceedings of the European Conference on Cognitive Ergonomics 2023,* 1-7. https://doi.org/10.1145/3605655.3605669

Hasselbring, W. & Reussner, R. (2006). Toward trustworthy software systems. *Computer*, 39(4), 91–92. https://doi.org/10.1109/MC.2006.142

Jaworski, M., & Piotrkowski, D. (2023). Study of software developers' experience using the GitHub Copilot Tool in the software development process. ArXiv, abs/2301.04991. https://doi.org/10.48550/arXiv.2301.04991

Kedar, S. (2024). *Tabnine vs. GitHub Copilot.* Tabnine. Retrieved on July 26, 2024, from https://www.tabnine.com/blog/tabnine-versus-GitHub-copilot/

Lipner, S. (2004). The trustworthy computing security development lifecycle. *Proceedings of the 20th Annual Computer Security Applications Conference,* 2–13. https://doi.org/10.1109/CSAC.2004.41

Lozano, I., Molina, J. & Gijón, C. (2021). Perception of artificial intelligence in Spain. *Telematics and Informatics*, 63. https://doi.org/10.1016/j.tele.2021.101672

Palacios-Gonzalez, E., Fernandez-Fernandez, H., Diaz, V., Garcia-Bustelo, B., & Lovelle, J. (2008). A review of Intelligent Software Development Tools*. Proceedings of the 2008 International Conference on Artificial Intelligence*, 585-590.

Shani, I., & GitHub Staff (2023). *Survey reveals AI's impact on the developer experience.* GitHub. Retrieved July 26, 2024, from https://GitHub.blog/2023-06-13-survey-reveals-ais-impact-on-the-developer-experience/

Snyk (2023). *2023 Snyk AI-Generated Code Security Report.* Snyk. Retrieved on July 26, 2024, from https://snyk.io/reports/ai-code-security/

Stack Overflow Retrieved July 26, from https://survey.stackoverflow.co/2020#overview

Treude, C., & Robillard, M. P. (2016). Augmenting API documentation with insights from stack overflow. *Proceedings of the 38th International Conference on Software Engineering,* 392-403. https://doi.org/10.1145/2884781.2884800

Wang, R., Cheng, R., Ford, D., & Zimmerman, T. (2024). Investigating and Designing for Trust in AI-powered Code Generation Tools. *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency, 1475-1493.* https://doi.org/10.1145/3630106.3658984

Zhang, B., Liang, P., Zhou, X., Ahmad, A., & Waseem, M. (2023). Demystifying Practices, Challenges and Expected Features of Using GitHub Copilot*. International Journal of Software Engineering and Knowledge Engineering*, 33(1) 1653-1672. https://doi.org/10.1142/S0218194023410048

ISSN: 2473-4901
v10 n6119

Appendix 1

Developer Type	Statistic	GitHub Copilot		
		Use	Do Not Use	Total
Academic Researcher	Count	294	1,060	1,354
	Percentage	**21.7%**	78.3%	
Blockchain	Count	122	201	323
	Percentage	37.8%	62.2%	
Cloud Infrastructure Engineer	Count	275	761	1,036
	Percentage	**26.5%**	73.5%	
Data or Business Analyst	Count	118	719	837
	Percentage	**14.1%**	85.9%	
Data Scientist or Machine Learning Specialist	Count	441	1,147	1,588
	Percentage	**27.8%**	72.2%	
Database Administrator	Count	34	223	257
	Percentage	**13.2%**	86.8%	
Designer	Count	44	237	281
	Percentage	**15.7%**	84.3%	
Developer Advocate	Count	74	138	212
	Percentage	**34.9%**	65.1%	
Developer Experience	Count	85	241	326
	Percentage	**26.1%**	73.9%	
Developer, Back-End	Count	3,123	10,622	13,745
	Percentage	**22.7%**	77.3%	
Developer, Desktop or Enterprise Apps. or Devices	Count	483	3,421	3,904
	Percentage	**12.4%**	87.6%	
Developer, Embedded Apps. or Devices	Count	215	1,630	1,845
	Percentage	**11.7%**	88.3%	
Developer, Front-End	Count	1,573	3,498	5,071
	Percentage	**31.0%**	69.0%	
Developer, Full-Stack	Count	7,582	18,153	25,735
	Percentage	**29.5%**	70.5%	
Developer, Game or Graphics	Count	161	705	866
	Percentage	**18.6%**	81.4%	
Developer, Mobile	Count	536	2,061	2,597
	Percentage	**20.6%**	79.4%	
Developer, QA or Test	Count	91	495	586
	Percentage	**15.5%**	84.5%	
DevOps Specialist	Count	355	1,032	1,387
	Percentage	**25.6%**	74.4%	

Developer Type	Statistic	GitHub Copilot		
		Use	Do Not Use	Total
Educator	Count	91	324	415
	Percentage	**21.9%**	78.1%	
Engineer, Data	Count	261	987	1,248
	Percentage	**20.9%**	79.1%	
Engineer, Site Reliability	Count	109	318	427
	Percentage	**25.5%**	74.5%	
Engineering Manager	Count	562	1,471	2,033
	Percentage	**27.6%**	72.4%	
Hardware Engineer	Count	29	257	286
	Percentage	**10.1%**	89.9%	
Marketing or Sales Professional	Count	25	124	149
	Percentage	**16.8%**	83.2%	
NA	Count	3,042	9,270	12,312
	Percentage	**24.7%**	75.3%	
Other	Count	585	2,495	3,080
	Percentage	**19.0%**	81.0%	
Product Manager	Count	89	357	446
	Percentage	**20.0%**	80.0%	
Project Manager	Count	91	498	589
	Percentage	**15.4%**	84.6%	
Research & Development Role	Count	254	1,099	1,353
	Percentage	**18.8%**	81.2%	
Scientist	Count	51	300	351
	Percentage	**14.5%**	85.5%	
Security Professional	Count	112	362	474
	Percentage	**23.6%**	76.4%	
Senior Executive	Count	486	846	1,332
	Percentage	**36.5%**	63.5%	
Student	Count	580	1,416	1,996
	Percentage	**29.1%**	70.9%	
System Administrator	Count	105	638	743
	Percentage	**14.1%**	85.9%	
Total	Count	22,078	67,106	89,184
	Percentage	**24.8%**	75.2%	