From Coders To Prompters; Evaluating Cognitive Impact Of The Usage Of Genai In Software Development - Developer Deskilling And Skill Atrophy

Nita Patil[1], Geeta Patil[2], Simarjyot Kaur Khanduja[3]
[1,2]*Asst Prof., Department of Computer Application, Marathwada Mitra Mandal's College of Commerce, Pune*
[3]*Student. MCA, Simarjyot Kaur Khanduja, DES Pune University*
doi.org/10.64643/IJIRT211713

Abstract*—Computer Science is integrated with emerging fields that include Generative AI (GenAI) and Large Language Models (LLMs) into each and every aspect in the modern times for software engineering. Almost all the software's have adopted these workflows. Software engineering has completely been reformed by including GenAI and LLMs, remodeling the computer science education. There has been enormous progress in the area of GenAI and nearly every software has incorporated GenAI and LLMs as a part of their software. Instances include GitHub, Copilot, Cursor, [2] Canva, VSCode Extensions, AI agents in specific apps, etc. Though the primary objective of instilling AI agents and systems in coding workflows was to enable convenience, productivity and programming speed of the end users and programmers, however, the prolonged effect of incorporation of these GenAI tools, Agents and LLMs in the coding software's on the junior developers is being neglected. The impact such inclusion has on the cognitive ability and logic building of Computer Science and engineering students is a much wider area of concern. This paper investigates the reconciliation of AI-powered productivity and human logic and skills spanning from students to industry ready entities utilizing an all- inclusive survey-based methodology that investigates how strong dependence and weak debugging capacities impact the ability of programmers on a long-term basis. The discoveries disclose the correlation between quick code implementation and deskilling of foundational logic building and core skill formation. Includes corporate and educational framework steps to be followed promoting a "developer-driven first, AI-driven later" approach for safeguarding the later innovations.

Index Terms*—GenAI, LLMs, Large language models, generative AI, coding, software engineering, AI-driven, Computer Science, GenerativeAI, coding softwares, coding platforms, coding, developing, platforms, deskilling, cognitive atrophy, skill atrophy.

I. INTRODUCTION

GenAI has influenced the computer science and engineering background in an enormous way.[1] Research suggests that GenAI is capable of generating solutions for computer science and engineering backgrounds coding assessments and code comprehension problems.[1] Professors voiced and articulated their issue that students might lean heavily on generative AI and use the agents and models to gain access to solutions generated by the GenAI without actually learning and gaining practical knowledge about the computer science concepts.[1]

Traditionally, the students had their own creativity and solutions to the assessments and academic problems, however, due to the advancement in technology and incorporation of generative AI in almost every sector, students have overlooked and neglected the importance of authenticity and efforts in assessments. For instance, earlier when computer science students used to get stuck on a line of code, they used to look up official documentations and websites like stack overflow to find out any syntactical or logical error and spend days on fixing the errors and bugs, on the contrary, in today's times, the programmers simply just prompt the AI agents for solutions and to fix bugs, even though they are minor ones.

Studies suggest that there has been a switch in the way the codes were written and maintained earlier.[2] It has also been researched that these large language models like Claude, ChatGPT, Gemini and Deepseek accomplish near pristine scores and set standards and measures for programming and many a times even exceed a wide number of human programmers in those

scores.[2] Generative AI has also enabled a new concept named "vibe coding" a way of programming which is used by coders where they cyclically coordinate with the large language models using prompts and acknowledge received code by those prompts without decoding and interpreting the logic and without perceiving the way the code is behaving. [2]

Exactly this is the point where the decision of incorporating GenAI and making students and professionals rely on it is something that should be questioned and given a second thought [2] because it can affect their logical and base building and could lead to cognitive atrophy.

Research also states that using large language models shifts the work from active logic building to passive assessing and reviewing work generated by AI.[2] This transition from active to passive is said to decrease efficiency of students and developers. [2]

The inclusion of GenAI in education systems has been discovered to raise some major concerns and challenges some of which include over reliance, inaccuracies, etc. [3] The researchers have mentioned that according to one of their anonymous methodology surveys, 54.5% of their students use Generative AI for solving assignment questions and that this emphasizes the threats of overreliance on the GenAI tools.[4]

Recent studies also suggest that over engagement with AI generated solutions leads to error prone solutions that leads to weak logical base and deskilling, it also directly affects productivity, it also suggests that over reliance on GenAI mitigates fulfillment obtained from logic building and error solving or debugging.[5]The main focus of mater is to identify whether the work is AI generated or done by the students because it raises questions on the university's integrity of the submissions of assessments.[6]

According to researchers conducted, large language models are said to generate solutions and provide code quickly and also provide solutions for errors, however, they are also said to have dangers which include SQL injections, security breaches, wrong output, incorrect logic which leads to long term suffer.[7]

According to an article, the code was structured, worked correctly, had no errors, in a reputed office, when asked about why it was organized in such a way the team just looked at the code for a few moments and replied that they used an AI agent to form the main pipeline of the code. This was concluding to one situation, the AI written code bypassed all the analysis, however no one could explain actually why was it formulated in such a way.[8] Studies state over reliance on the LLMs could lead to bugs in critical industry level software's and also the issues would be time consuming.[9] Researchers mention that depending wholly on GenAI leads to incremental and slow skill erosion of problem-solving skills and basic core foundational knowledge.[10]

II. OBJECTIVES:

1. The primary objective of our research paper includes identifying the impact of depending overly on GenAI and LLMs on cognitive ability and logic building and problem-solving skills.
2. Our research paper, on the other hand, also aims to evaluate the logical errors, attempts of SQL injections, time wastage in debugging erroneous code copied from GenAI.
3. Additionally, another aspect of focus in the paper includes the compromise of logical base building and cognitive atrophy, just for saving time to build projects.
4. Our objective in the paper also aims to cover guidelines for educational entities and professional entities to adopt the no GenAI or zero GenAI days in between their schedules to enhance skill building amongst peers and also having them instilled with the habit of *"developer-driven first, AI-driven later"* approach.

III. LITERATURE REVIEW

The coding fields are now getting swapped as prompting fields because of the rapid inclusion of GenAI for programming. Programming tools prompt and complete the whole code for the programmers without even them having to write a single line of code. This causes the programmer's logical base to decline and leads to skill atrophy. The inclusion of LLMs in the field of computer science for coders and developers leads to skill erosion.

IV. METHODOLOGY:

The data utilized in our research paper was collected by administering an online survey, targeting programmers, Computer science background students

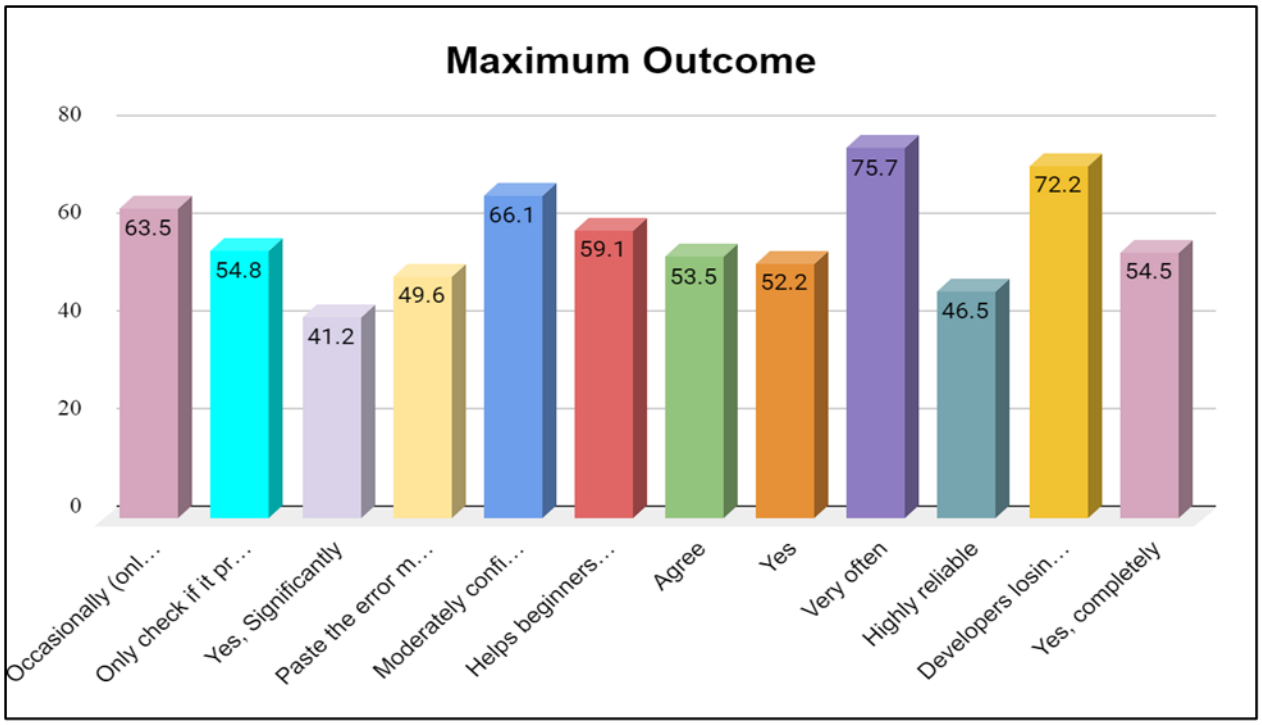
and freelancers. The analysis was performed using Google Forms, the data collected is quantitative in nature. Real world, closed-ended questions were given in the form. The accumulated data was then analyzed and added in the research paper.

The survey was conducted for working professionals and students, age group 18 to 60 for gathering data concerning the usage of GenAI and LLMs in the field of Computer Science. The Conducted survey had a response of 107 people. The data was collected, analyzed and later added to the contents section with the outcomes. The result of all the findings is stated below in a tabular format and includes the result summary.

V. ANALYSIS PLAN / DISCUSSIONS

Table 1: Table-1 shows maximum outcome of survey.

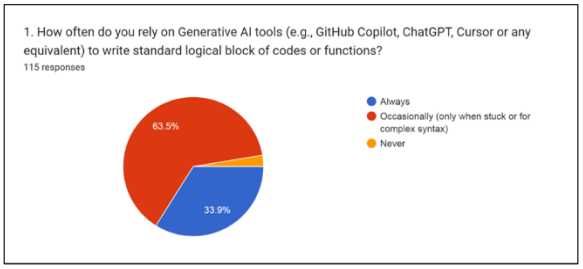
Questions	Maximum Outcome	Percentage
1. How often do you rely on Generative AI tools (e.g., GitHub Copilot, ChatGPT, Cursor or any equivalent) to write standard logical block of codes or functions?	Occasionally (only when stuck or for complex syntax)	63.5
2. When an AI code assistant gives a working solution, how intensively do you read and understand every part before integrating it?	Only check if it provides the expected output	54.8
3. Do you think your proficiency to write algorithms, syntax, or data structures from scratch (without internet or AI assistance) has reduced over the years?	Yes, Significantly	41.2
4. When coming across a runtime error or complex bug, what immediate first step does you follow?	Paste the error message directly to an AI prompt for a quick fix.	49.6
5. How confident are you in debugging a huge unfamiliar block of code if all AI coding assistants were temporarily unavailable?	Moderately confident	66.1
6. How does early reliance on AI to generate code affect the foundation of problem-solving and logic building?	Helps beginners to build quick projects	59.1
7. Should universities and tech companies constrain AI-free coding challenges to prevent Deskilling?	Agree	53.5
8. Have you ever uncovered hidden security traps (e.g., hardcoded secrets, SQL injection, weak authentication, or unsafe dependency) in AI-generated code?	Yes	52.2
9. How frequently does the time saved writing code with AI get wasted later while debugging the logic failures?	Very often	75.7
10. In your experience, how reliable are AI code generators when working with complex system architecture?	Highly reliable	46.5
11. In Aggregation for the long run, what is your biggest concern about heavy AI coding use?	Developers losing their core logical and problem-solving skills	72.2
12. Do you believe in AI-generated code enough to use it in huge, complex and live projects?	Yes, completely	54.5



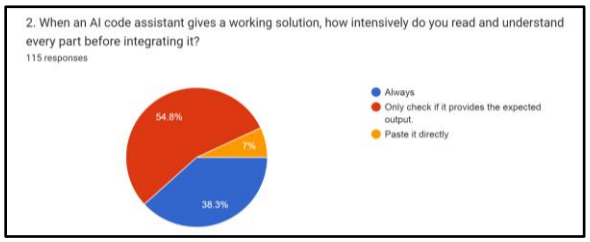
Graph 1: Graph-1 shows graphical representation of maximum outcome of survey.

VI. RESULTS AND FINDINGS

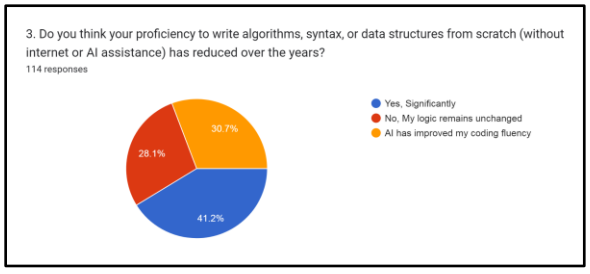
1. In accordance with the survey carried out, when the programmers were asked, how often do they rely on AI tools, 33.9% of responders said they used it always and 63.5% said they used AI tools occasionally (only when stuck).



2. When asked “When an AI code assistant gives a working solution, how intensively do you read and understand every part before integrating it?” 54.8% of responses said they only check if it provides the expected solution and use it directly. This clearly states how coders and programmers just check for and validate the solution given by AI tools to complete their intended work without even understanding what actually is happening and why it is written that way.

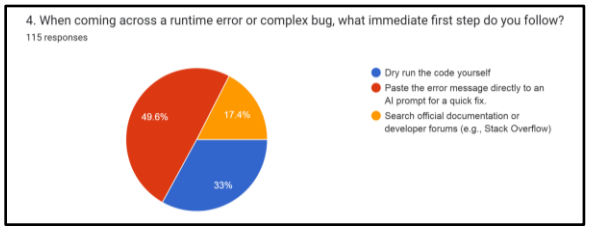


3. When the respondents were questioned whether they think their proficiency to write codes and Data structures without internet or AI tools has reduced, 41.2% respondents felt it has done so significantly.

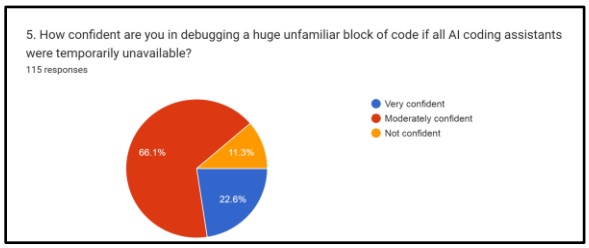


4. According to the analysis on the survey, when the responders were asked how they fix runtime errors, or complex bugs, 49.6% of responders selected the option that they directly paste the error message to AI Prompt for quick fixes. This states that coders have nowadays transitioned and become prompters because

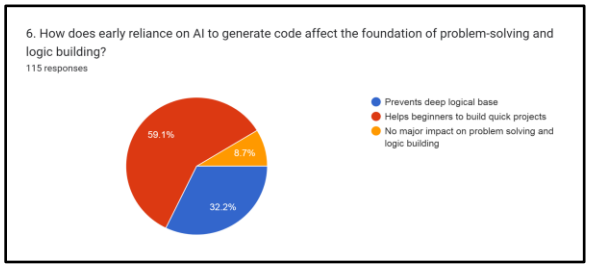
the majority of them do not even try to read or understand the errors. The earlier methods of looking up documentations are near to eradication because only 17.4% say they look up the official documentation.



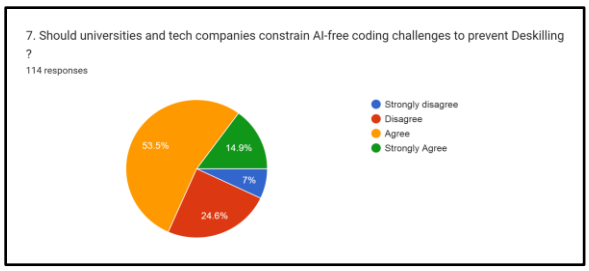
5. When asked how confident programmers and professionals felt to debug unfamiliar blocks of code, 66.1% said they were moderately confident, whereas debugging is one of the core skills in computer science background and they are supposed to be very confident about it.
Only 22.6% say they are very confident about the same.



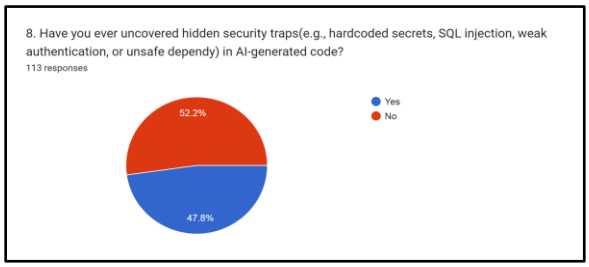
6. 59.1% of Coders and programmers feel that early and over reliance on AI helps beginners build quick projects, whereas 32.2% feel it prevents a deep logical base.



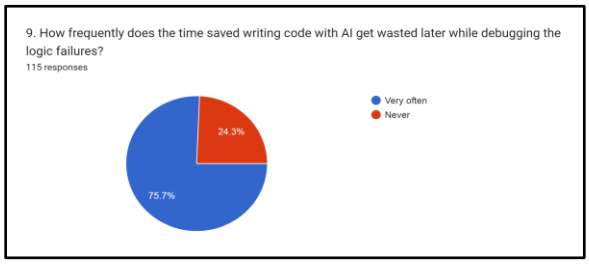
7. According to the survey,14.9% strongly agree and 53.5% of Professionals, students and coders agree to organize AI free coding challenges to prevent cognitive atrophy and deskilling.



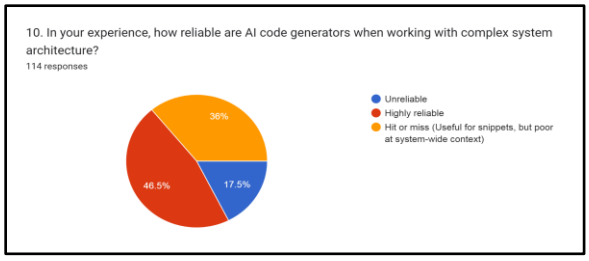
8. 47.8% of responders said they uncovered hidden security breaches and traps in the AI - generated code. This raises safety concerns and explains why GenAI code should not directly be copy pasted.



9. 75.5% of people say when code is copy pasted from genAI, the time saved in writing code later gets wasted in debugging the same code generated by the AI tools.

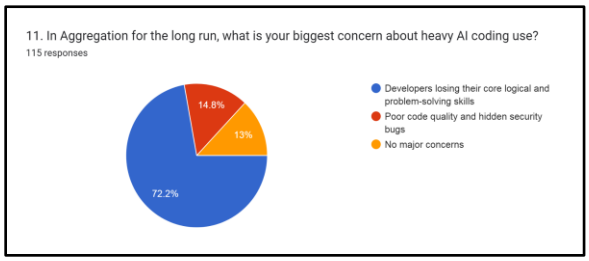


10. The survey analysis also concludes that 36% of users feel the AI generated code is a Hit or miss for complex architectures, while 17.5% feel it is highly unreliable which concludes to 53.5% people feeling that AI generated code is not relevant in complex architectures.

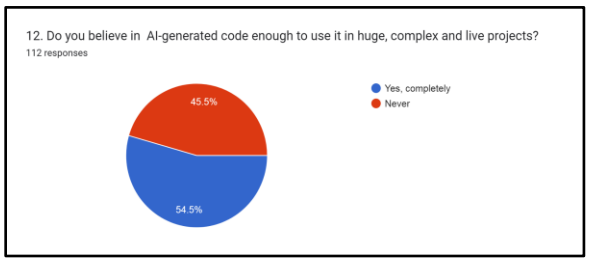


11. To understand how the programmers and coding professionals feel about the long run impact of heavy

usage of AI tools, the question “What is the biggest concern about heavy AI usage” was asked and 72.2% of responders feel that the developers will lose their core logical and problem-solving skills.



12. 45.5% of responders also feel that AI Generated code is never to be believed on in live projects and live case scenarios.



VII. CONCLUSION

In accordance with the results and findings gained by the conducted survey for this research paper, analysis describes the tradeoffs between short term productiveness and long term cognitive and skill atrophy. The survey shows that relying heavily on AI systems leads to cognitive deskilling, logical deskilling and how coders are transiting from being coders to prompters.

To avoid all of these situations to occur, the tech-based companies and the computer science and engineering backgrounds should timely and recursively implement “Zero AI” or “No AI” coding challenges for programmers and students to prevent deskilling, to build logical base and to improve problem-solving among them. Also, they should emphasize “developer-driven first, AI-driven later” approach for the betterment in logic building and to avoid cognitive atrophy. The generation should always remember that AI was designed to magnify and intensify human intelligence and not to swap roles with human proficiency.

REFERENCES

[1] T. H. Feng, A. Luxton-Reilly, B. C. Wünsche, and P. Denny, “From automation to cognition: Redefining the roles of educators and generative AI in computing education,” University of Auckland, Auckland, New Zealand. doi: 10.1145/3716640.3716658. [Online]. Available: https://dl.acm.org/doi/full/10.1145/3716640.3716658

[2] S. Maier, M. Gunzenhäuser, J. Schweisthal, M. Schneider, and S. Feuerriegel, “A meta-analysis of the effect of generative AI on productivity and learning in programming,” arXiv preprint arXiv:2605.04779, 2026. [Online]. Available: https://arxiv.org/pdf/2605.04779

[3] A. S. Yalwa, M. S. Othman, L. M. Yusuf, and M. S. Almarshadi, “Generative AI in higher education: A systematic review with emphasis on programming and computer science education.” [Online]. Available: https://www.researchgate.net/profile/Abdulaziz-Saidu-Yalwa/publication/406084437_Generative_AI_in_Higher_Education_A_Systematic_Review_with_Emphasis_on_Programming_and_Computer_Science_Education/links/6a27e3aed551e312d71f8a7b/Generative-AI-in-Higher-Education-A-Systematic-Review-with-Emphasis-on-Programming-and-Computer-Science-Education.pdf

[4] E. Dickey, A. Bejarano, and C. Garg, “AI-Lab: A framework for introducing generative artificial intelligence tools in computer programming courses,” *SN Computer Science*, vol. 5, Art. no. 720, 2024, doi: 10.1007/s42979-024-03074-y.

[5] W. Yin, “The impact of generative artificial intelligence on college students’ computer thinking in the task of complex computer programming: Based on social cognitive theory,” *Creative Education*. [Online]. Available: https://www.scirp.net/pdf/ce_6309199.pdf

[6] Z. Ahmed *et al.*, “The generative AI landscape in education: Mapping the terrain of opportunities, challenges, and student perception,” *IEEE Access*, vol. 12, pp. 147023–147050, 2024, doi: 10.1109/ACCESS.2024.3461874.

[7] B. C. Matias, S. Freire, J. Freitas, F. Fronchetti, K. Damevski, and R. Spinola, “A survey on large

language model impact on software evolvability and maintainability: The good, the bad, the ugly, and the remedy,” arXiv preprint arXiv:2601.20879, 2026. [Online]. Available: https://arxiv.org/html/2601.20879v1

[8] “AI wrote the code. Tests pass. But nobody can explain why it was built this way: Comprehension debt—The hidden cost of AI-generated code,” *Medium*. [Online]. Available: https://medium.com/predict/ai-wrote-the-code-tests-pass-but-nobody-can-explain-why-it-was-built-this-way-7c4baa27844a

[9] K. Kompella, “The risks of AI overreliance in software engineering,” *TechTarget*, Oct. 5, 2026. [Online]. Available: https://www.techtarget.com/news/The-risks-of-AI-overreliance-in-software-engineering/833966/

[10] L. Kinyanjui, “The disadvantages of depending wholly on AI tools for software development,” *Medium*. [Online]. Available: https://medium.com/@LucieKinyanjui/the-disadvantages-of-depending-wholly-on-ai-tools-for-software-development-49189937f557