

Evaluating the User Experience of AI-Assisted Programming Tools among Computer Studies College Students

¹Mark Andrew D. Ancheta., ¹Raizelle M. Diamante., ¹Precy Mae L. Egloso., ²Dr. Erlito M. Albina

¹School of Computer Studies, Arellano University – JAS Campus Pasay City, Philippines

²School of Computer Studies, Lyceum of the Philippines University Pasay, Philippines

DOI: <https://doi.org/10.51244/IJRSI.2026.1313CS033>

Received: 08 October 2026; Accepted: 13 October 2026; Published: 26 October 2026

ABSTRACT

This study examined the perceived user experience of AI-assisted programming tools among 60 respondents in the National Capital Region, composed of Computer Studies college students, recent graduates, and IT professionals, using a quantitative descriptive design. Most participants were beginner to intermediate BSIT students who use platforms such as ChatGPT several times a week to explain code, learn concepts, and fix bugs. Respondents generally evaluated the tools positively, giving an overall weighted mean of 3.20 (Agree). Usefulness and Productivity received the highest rating (3.28, Strongly Agree), while Code Quality and Accuracy received the lowest (3.08, Agree). Respondents perceived the tools as helpful supports that make work faster, but they also expressed reservations about the reliability and accuracy of AI-generated code. The study recommends that school guide students in double-checking the suggestions of AI-assisted programming tools.

Keywords: Artificial Intelligence, AI-assisted programming tools, user experience, programming education, Computer Studies students

INTRODUCTION

Artificial intelligence is rapidly changing how computer science is taught, as schools now use coding assistants such as ChatGPT, GitHub Copilot, and Google Gemini. Although these tools make it easier to program by producing code and immediately correcting bugs, it is still necessary to carry out thorough practice and apply critical thinking in order to master the subject. AI presents a double-edged situation in that it lessens coding anxiety and increases the speed of work, but an excessive dependence on it can impair the ability to solve problems independently.

This is a significant challenge for college students studying Computer Studies in the National Capital Region (NCR), as they must balance using AI with developing real software skills. Although existing studies point out advantages such as increased confidence, most of the research has been carried out in Western contexts and not among Filipino students. In the Philippines, initial studies have examined AI tool adoption among Computer Studies students in Quezon City [1], the impact of AI-powered tools on the programming skills of IT students in Davao Oriental [2], and ChatGPT integration in undergraduate IT education programs nationwide [3]. However, none has described the user experience of AI-assisted programming tools across different universities in the National Capital Region, which is the gap this study addresses.

LITERATURE REVIEW

Students' willingness to use AI coding assistants is shaped by trust, motivation, perceived usefulness, and academic goals [4]. Ancog et al. (2026) identify perceived usefulness as a key adoption driver, with students viewing ChatGPT and GitHub Copilot as learning aids [1]; however, concerns over plagiarism, privacy, dependency, and limited school support make usage inconsistent [1]. Farooq et al. (2026) explain that whether students adopt these tools depends on a mix of personal, school-related, and technology-based factors [5]. Liang et al. (2024) observed that learners turn to AI to speed up their work and get syntax right, though they often

reject code that doesn't fit what they need [6]. Meanwhile, Xu et al. (2022) point out that smooth integration into coding platforms and easy-to-understand language are key to keeping students using these tools over time

[7]. Barke et al. (2023) distinguish "acceleration" and "exploration" modes [8]; Mozannar et al. (2022)'s CUPS framework shows efficiency depends on both generation speed and verification time [9]. Preferences vary by task, prioritizing accuracy and conversational responses [8], [10].

Impacts on learning are mixed. Wermelinger (2023) cites benefits in explanation and debugging but warns of integrity risks and eroding independence [11]. Beginners mostly use AI for guidance, not full code [3]; combining AI with manual coding yields best outcomes, while over-reliance does not [12]. Structured AI integration and collaboration strengthen critical thinking and resilience - benefits not guaranteed by access alone [5], [13]. Guided use aids novices; unstructured dependence hinders skill growth [14]. Faculty note success depends on institutional and pedagogical support, not just availability [15]. Outcomes vary: gains and skill erosion are both documented [16]. Ready-made answers can reduce self-reliance [17]; satisfaction rises with clear, step-by-step guidance and alignment with academic needs [18] [19] [20] [21], yet may coexist with reduced independence, weaker conceptual grasp, and lingering accuracy doubts [9] [22] [23]. Philippine studies confirm AI's potential for personalized learning alongside risks of unreliable outputs, over-reliance, and integrity issues - plus a need for sound design and institutional backing [1] [2], [24]. Overall, experience stems from usefulness, usability, trust, satisfaction, and context; most research draws on Western settings or professionals, and no study has focused on Computer Studies students in the National Capital Region, a gap this study addresses.

METHODOLOGY

A. Research Design

The study uses a quantitative descriptive research design to examine Computer Studies students' experiences with AI-assisted programming tools. Numerical survey responses are used to describe participants' assessments of usefulness, usability, code quality, learning facilitation, satisfaction, confidence, and accuracy. The design summarizes reported experiences and perceptions; it does not test a causal effect of AI-tool use.

B. Respondents

The respondents are individuals in the National Capital Region (NCR) with experience using AI-assisted programming tools. They include first- to fourth-year Computer Studies college students, recent graduates, and IT professionals. The study aimed to gather 60 valid responses, with student participation across year levels. Recent graduates and IT professionals were deliberately included because they come from the same Computer Studies background that current students are completing. Having finished the same academic tasks, they can show whether the experiences reported by current students carry over into actual practice, adding experience levels to the sample while keeping the focus on Computer Studies backgrounds in the NCR (52 students, 86.7%; five recent graduates, 8.3%; three IT professionals, 5.0%).

C. Sampling Design

Purposive sampling, a non-probability sampling method, was used to recruit eligible respondents with prior experience using AI-assisted programming tools. The online questionnaire remained open until 60 valid responses were obtained.

D. Research Instrument and Survey Questionnaire

Questionnaire and Response Scale

The primary instrument is a structured online questionnaire administered through Google Forms. The instrument is divided into two major sections: Part 1 gathers respondents' demographic information and AI tool usage patterns, while the evaluation sections contains 20 evaluation statements in five categories - Part A, Usefulness and Productivity (4 items); Part B, Usability and Accessibility (4 items); Part C, Code Quality and Accuracy (4

items); Part D, Learning Facilitation (4 items); and Part E, Programming Performance and Satisfaction (4 items) - each rated on a four-point Likert scale.

1 - Strongly Disagree

2 – Disagree

3 – Agree

4 - Strongly Agree

Existing AI Programming Tools

The study considers commonly used AI programming tools reported by respondents, including ChatGPT, Google Gemini, Microsoft Copilot, and GitHub Copilot



Figure 1. User Interface Prototype of AI-Assisted Programming Tool

Figure 1 presents the study's interface prototype. It is included only to illustrate the interface that existing AI-assisted programming tools commonly have: a navigation sidebar for chat history, a main workspace for code or images generated from prompts, and a context panel for file structures and live comparisons of changes. The prototype was not developed, deployed, or evaluated in this study; the survey items refer to respondents' experiences with existing AI programming tools in general. Testing the prototype separately through a clearly described evaluation procedure is recommended for future work.

Instrument Validation

Before administration, the research adviser reviewed the questionnaire to check that the statements were clear and relevant, selected students tried it out to identify confusing items, and revisions were made when needed. After data gathering, the internal consistency of the instrument was tested using Cronbach's alpha computed from the responses of the 60 valid respondents: the full 20-item instrument obtained an alpha of 0.944 (excellent), and the category coefficients were 0.880 for Usefulness and Productivity, 0.830 for Usability and Accessibility, 0.810 for Code Quality and Accuracy, 0.795 for Learning Facilitation, and 0.848 for Programming Performance and Satisfaction, all within the acceptable to good range (0.70 and above).

E. Data-Gathering Procedure

The questionnaire was developed from the study objectives and variables related to AI programming-tool use. After review by the research adviser, it was prepared in Google Forms and distributed to eligible respondents in the NCR. Participants were informed of the study's purpose and asked to answer honestly based on their own experiences. Data collection continued until 60 valid responses had been obtained. The responses were then compiled and prepared for statistical analysis.

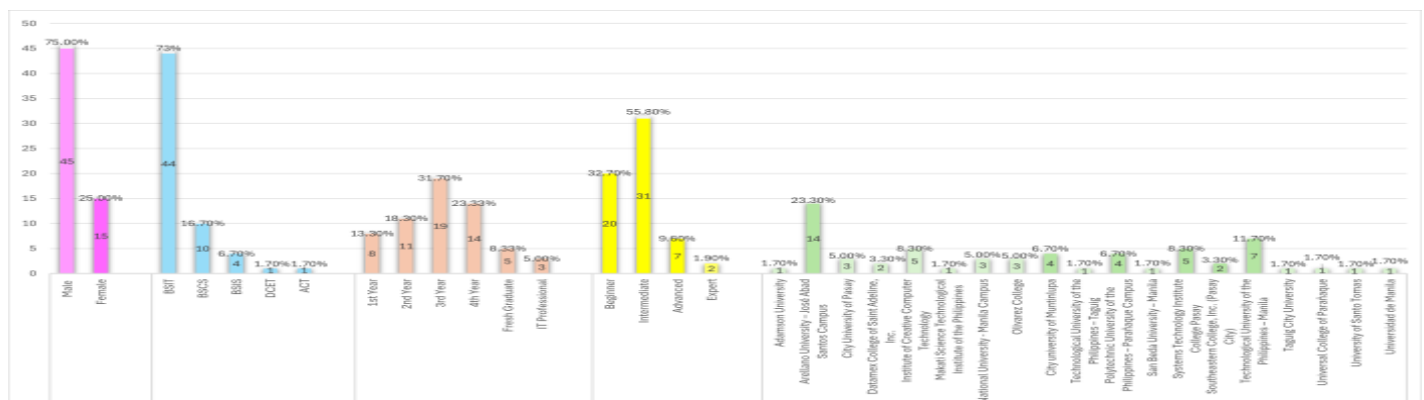
F. Statistical Treatment of Data

Frequency and percentage are used to summarize respondents' demographic profiles and AI-tool usage patterns. Percentage is calculated as $P = (f / N) \times 100$, where f is the frequency of responses and N is the number of valid respondents. The weighted mean is used to summarize ratings for each evaluation statement and user-experience area: $WM = \Sigma(fw) / N$, where f is the number of responses at a rating, w is the numerical weight assigned to that rating, and N is the number of valid responses. The standard deviation is used to describe the variation in responses. For interpreting weighted means, 3.26–4.00 corresponds to Strongly Agree, 2.51–3.25 to Agree, 1.76–2.50 to Disagree, and 1.00–1.75 to Strongly Disagree. These ranges are for interpreting the computed means; the questionnaire response options remain 1 - Strongly Disagree, 2 - Disagree, 3 - Agree, and 4 - Strongly Agree. The internal consistency of the instrument was assessed using Cronbach's alpha: $\alpha = (k / (k - 1)) \times (1 - (\Sigma\sigma_i^2 / \sigma_t^2))$, where k is the number of items, $\Sigma\sigma_i^2$ is the sum of the item variances, and σ_t^2 is the variance of the total scores across respondents.

RESULTS

This chapter presents the results gathered from 60 respondents, composed of Computer Studies College students, fresh graduates, and IT professionals, regarding their experience with AI-assisted programming tools. The respondents were asked to provide information about their demographic profile and usage patterns, as well as their assessment of the tools in terms of usefulness, usability, code quality, learning facilitation, and overall satisfaction.

Chart 1. Demographic Profile Of The Respondents (n = 60)



The majority of the respondents were male, constituting 75.0% (45) while only 25.0% (15) were female. Most of the respondents were Bachelor of Science in Information Technology students which comprised 73.3% (44). The other students were from BSCS comprising 16.7% (10) and BSIS consisting 6.7% (4). Only one respondent each came from ACT and DCET programs. In terms of academic status, third-year students constituted the highest percentage amounting to 31.7% (19). This was followed by fourth-year students consisting 23.3% (14), second-year students comprising 18.3% (11) and first-year students making up 13.3% (8). In addition, there were five fresh graduates (8.3%) and three working IT professionals (5.0%). In terms of programming proficiency, more than half of the respondents identified themselves as Beginners which amounted to 51.7% (31). This was followed by Intermediate consisting 33.3% (20), Advanced comprising 11.7% (7) and Expert which only comprised 3.3% (2). In terms of school or university, the largest group of respondents came from Arellano University – José Abad Santos Campus at 23.3% (14). This was followed by Technological University of the Philippines – Manila with 11.7% (7), Systems Technology Institute College Pasay and Institute of Creative Computer Technology Colleges both at 8.3% (5), and City university of Muntinlupa and Polytechnic University of the Philippines – Parañaque Campus each at 6.7% (4). National University - Manila Campus, Olivarez College, and City University of Pasay all had 5.0% (3), while Datamex College of Saint Adeline, Inc. and Southeastern College each had 3.3% (2). Adamson University, Makati Science Technological Institute of the Philippines, Technological University of the Philippines – Taguig, San Beda University Manila, Taguig City University, Universal College of Parañaque, University of Santo Tomas, and Universidad De Manila each had 1.7% (1) respondent. All of these figures are based on respondents' self-reported information.

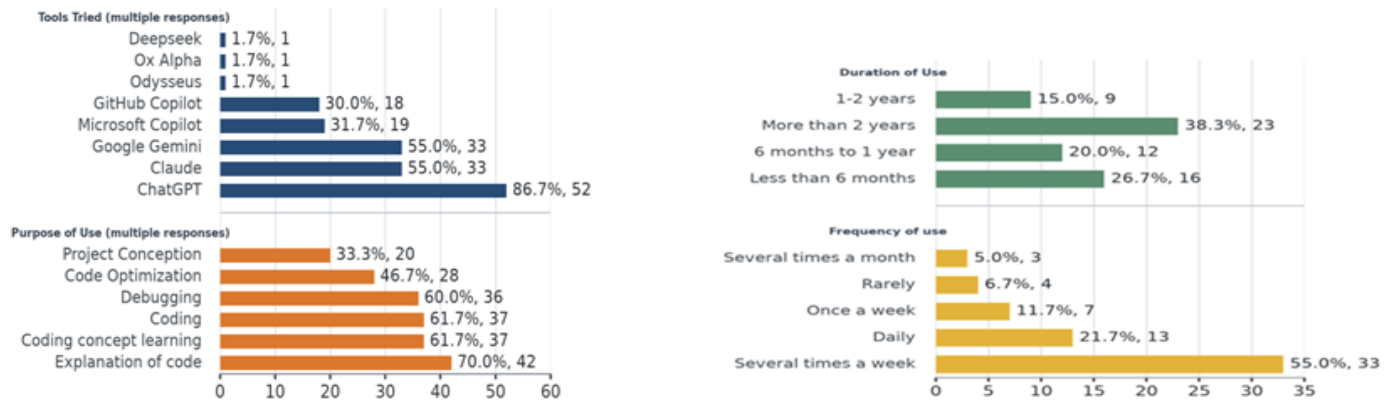


Chart 2 and 3. Usage Patterns Of Ai Programming Tools

In terms of the frequency of the usage, more than a half of participants (55.0%) use AI programming tools several times a week, 21.7% use them daily. As for the time of use, 38.3% of participants have used such tools for more than two years, 26.7% have used them for less than six months, 20.0% – for six months to one year and 15.0% – for one to two years. The most popular platforms, according to participants are ChatGPT (86.7%), followed by Claude and Google Gemini (55.0% each), Microsoft Copilot (31.7%) and GitHub Copilot (30.0%). Finally, the most common purposes include code explanation (70.0%), coding concept learning (61.7%), general coding (61.7%), and debugging (60.0%). Such a prevalence of code explanation and debugging corresponds to the results of the previous studies, which found that novice programmers mostly rely on AI assistants for learning concepts, explaining code, and debugging rather than coding [12], [20], [25].

Summary of Categories Means

Table 1

Categories	Weighted Mean	Interpretation
Usefulness and Productivity	3.28	Strongly Agree
Usability and Accessibility	3.20	Agree
Code and Accuracy	3.08	Agree
Learning Facilitation	3.22	Agree
Programming Performance and Satisfaction	3.23	Agree
Average Mean:	3.20	Agree

Table 1 shows the overall assessment scores of the five item categories. Given the generally positive ratings, it can be concluded that Computer Studies students find AI-assisted programming tools mostly beneficial. Usefulness and Productivity received the highest assessment (3.28, Strongly Agree), indicating that students perceived that they could complete their programming tasks faster and with better results using the tools. Programming Performance and Satisfaction (3.23) and Learning Facilitation (3.22) suggest that, with the tools' help, students accomplish tasks with greater confidence and develop a better understanding of the material. Usability and Accessibility (3.20) indicate that the tools are generally easy to use and accessible. Code Quality and Accuracy (3.08) received the lowest rating among the five categories, revealing students' reservations about the reliability of AI-generated code and the continued need to check, verify, and debug the code produced by these tools. This result mirrors earlier findings that users often reject AI suggestions when the generated code fails to meet their requirements [6] and that accuracy concerns remain even among satisfied student users [3],

[22]. Because these ratings are self-reported perceptions, they describe how respondents experienced the tools rather than directly measured speed, productivity, or programming performance.

Taken together, the results suggest that students value AI-assisted programming tools mainly as supports for code explanation, concept learning, and debugging, which are the very purposes they reported using most often, rather than as replacements for their own coding. The combination of high usefulness ratings and the lowest accuracy rating indicates that a positive user experience does not necessarily mean full trust in the tool's output, which supports the need for institutional guidance on responsible and verified AI use [1] [3]

SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

A. Summary of Findings

The survey of 60 Computer Studies college students, fresh graduates, and IT professionals in the National Capital Region showed that the majority of respondents were male (75.0%), BSIT students (73.3%), and third-year students (31.7%), with more than half (51.7%) rating their programming proficiency as Beginner. Most respondents use AI-assisted programming tools several times a week (55.0%), with 38.3% using them for more than two years, and ChatGPT was the most used tool (86.7%). The top purposes of use were code explanation (70.0%), coding concept learning (61.7%), and debugging (60.0%). The overall weighted mean of 3.20, interpreted as Agree, reflects a generally positive user experience, with Usefulness and Productivity rated highest (3.28) and Code Quality and Accuracy rated lowest (3.08).

B. Conclusion

The study concludes that AI-assisted programming tools provide a generally favorable user experience among Computer Studies students in the National Capital Region. The tools are valued mainly as supports for code explanation, concept learning, and debugging rather than as replacements for independent programming. However, because code quality and accuracy received the lowest rating, even satisfied users remain cautious about the correctness of AI-generated code. The benefits of these tools therefore depend on responsible and guided use, and their integration into programming education should preserve students' independent problem-solving skills.

Limitations of the Study

This study has several limitations. The 60 respondents were recruited through purposive sampling from a limited set of universities in the National Capital Region, so the findings cannot be generalized to all Computer Studies students in the Philippines. All data were self-reported perceptions; no direct measurements of coding speed, productivity, or programming performance were conducted. The descriptive, one-time design does not establish causal effects of AI-tool use on learning outcomes. Respondents' prior exposure to AI tools ranged from less than six months to more than two years, which may have influenced their assessments. Finally, the interface prototype served only as an illustration of features commonly shared by existing tools and was not itself evaluated.

C. Recommendations

Based on the results and conclusion, recommendations can be done as:

1. *For students* - They should use AI tools for code explanation, learning purposes, or debugging. Also, it is needed to check, debug, and validate suggestions provided by such sources before submitting them.
2. *For educators and the institution* - There is a need to provide students with training about the opportunities and threats of AI and specific requirements for utilizing such sources for programming tasks. Also, it is needed to create assignments where students should explain, debug, and test the code suggested by AI sources to prevent its careless implementation.

3. *For future researchers* - Conduct a follow-up study with a larger sample size covering other regions and programs, and consider qualitative or experimental designs to examine how specific tool features affect accuracy concerns and learning outcomes.

REFERENCES

1. C. I. Ancog, H. . R. Lucero, I. . R. Estrada, H. A. T. Longaquit and N. . T. Mayo, "Factors Influencing AI Tool Adoption in Research Among Junior and Senior Students of QCU College of Computer Studies," *International Journal of Computing Sciences Research*, vol. 15, pp. 1313-1326, 6 2026.
2. D. C. Montejo, K. A. B. Diocares, J. C. Arac, K. H. S. Adol and S. G. Mendoza, "Evaluating the impact of AI-powered tools on programming skills development among IT students at Davao Oriental State University (DOrSU)," *HO CHI MINH CITY OPEN UNIVERSITY JOURNAL OF SCIENCE - SOCIAL SCIENCES*, vol. 16, no. 2, pp. 98-114, 10 2026.
3. I. M. Cagula-Escauso, R. . B. Ardael-Gabayno, J. Casido, . N. Del Rosario and J. . L. Plender-Nabas, "Impact of ChatGPT Integration on Programming Skill Development in Undergraduate IT Education Programs in the Philippines: A Systematic Review," *SSRN*, 2026.
4. F. Ali, A. Ahmed, M. A. Alipour and H. Terashima-Marin,, "Adoption of AI-coding assistants in programming education: exploring trust and learning motivation through an extended technology acceptance model," 2026.
5. U. Farooq, D. Morganti and S. Anwar, "AI Code Assistants in Programming Education: A Narrative Literature Review," *Education Sciences*, vol. 16, no. 6, p. 961, 6 2026.
6. J. T. Liang, C. Yang and B. A. Myers, "A Large-Scale Survey on the Usability of AI Programming Assistants: Successes and Challenges," in *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering (ICSE '24)*, 2024.
7. F. F. Xu, . V. Bogdan and G. Neubig, "In-IDE Code Generation from Natural Language: Promise and Challenges," 4 3 2022.
8. S. Barke, M. B. James and N. Polikarpova, "Grounded Copilot: How Programmers Interact with Code-Generating Models," *Proceedings of the ACM on Programming Languages*, vol. 7, no. OOPSLA1, p. 85–111 (Article 78), 6 4 2023.
9. H. Mozannar, A. Fourney, . G. Bansal and . E. Horvitz, "Reading Between the Lines: Modeling User Behavior and Costs in AI-Assisted Programming," *arXiv*, 2022.
10. S. . I. Ross, F. Martinez, S. Houde, M. Muller and J. D. Weisz, "The Programmer's Assistant: Conversational Interaction with a Large Language Model for Software Development," 27 3 2023.
11. M. Wermelinger, "Using GitHub Copilot to Solve Simple Programming Problems," in *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE '23)*, 2023.
12. I. M. Samarappulige, K. Ekanayake and N. T. Jayathilake, "Evaluating the Educational Benefits and Risks of AI Coding Assistants Among Novice Programming Students in Sri Lanka," *TechRxiv*, 2025.
13. W. Zheng, T. Ratanaolarn and K. Sriwisathiyakun, "A quasi-experimental study of lab rotation integrated with generative AI and peer code review model to enhance students' critical thinking and academic resilience in introductory programming," *Cogent Education*, vol. 13, no. 1, 2026.
14. M. Alanazi, A. Li, . A. Almalawi, H. Samra and . B. Soh, "Pedagogical Transformation and Teaching Practice in Programming Education Through AI Coding Assistants: Faculty Perspectives and the AI Coding Assistant Adoption Framework," *Applied Sciences*, vol. 16, no. 10, p. 4833, 5 2026.
15. F. Nizamudeen, Gatti, Lorenzo, Bouali, Nacir and Ahmed, Faizan, "Investigating the Impact of Code Generation Tools (ChatGPT & Github CoPilot) on Programming Education," in *Proceedings of the 16th International Conference on Computer Supported Education (CSEDU 2024)*, 2024.
16. T. Kosar, D. Ostojić, Y. D. Liu and M. Mernik, "Computer Science Education in ChatGPT Era: Experiences from an Experiment in a Programming Course for Novice Programmers," *Mathematics*, vol. 12, no. 5, p. 629, 21 2 2024.
17. L. P. Plecerda, "Academic integrity surrounding the use of generative AI in higher education: Lenses from ICT college students," *International Journal of Higher Education*, 30 12 24.
18. M. Kazemitabaar, X. Hou, A. Henley, B. J. Ericson, D. Weintrop and T. Grossman, "How Novices Use LLM-based Code Generators to Solve CS1 Coding Tasks in a Self-Paced Learning Environment," in

Proceedings of the 23rd Koli Calling International Conference on Computing Education Research (Koli Calling '23), 2023.

19. C. Rahe and W. Maalej, "How Do Programming Students Use Generative AI?," Proceedings of the ACM on Software Engineering, vol. 2, pp. 978- 1000, 6 2025.
20. R. Zvieli-Girshin, "The Good and Bad of AI Tools in Novice Programming Education," Education Sciences, p. 1089, 6 10 2024.
21. J. M. M. Fatagani-Valerio, . J. R. G. Valerio, B. . P. Balancio and . J. L. Pimentel, "An Assessment of Student-Researcher Satisfaction with the Use of Artificial Intelligence in Thesis Writing," Psychology and Education: A Multidisciplinary Journal, vol. 46, no. 7, p. 920–926, 2025.
22. B. Sheese, M. Liffiton, J. Savelka and P. Denny, "Patterns of Student Help-Seeking When Using a Large Language Model-Powered Programming Assistant," in Proceedings of the 26th Australasian Computing Education Conference (ACE '24), 2024.
23. P. Denny, S. MacNeil, J. Savelka and A. Luxton-Reilly, "Desirable Characteristics for AI Teaching Assistants in Programming Education," in Proceedings of the 2024 Innovation and Technology in Computer Science Education Conference V. 1 (ITiCSE '24), 2024.
24. R. M. Llobia and . D. R. . B. Pasion, "Generative Artificial Intelligence Integrated Pedagogy and Quality of Student Output in Web Design Courses Among Information Technology," International Journal of Science, Architecture, Technology, and Environment, vol. 3, no. 5, 5 5 2026.
25. E. Chen, . R. Huang, H. S. Chen, . Y. H. Tseng and L. Y. Li, "GPTutor: A ChatGPT-Powered Programming Tool for Code Explanation," in Proceedings of the 24th International Conference on Artificial Intelligence in Education (AIED 2023), 2023.