Jurnal Teknologi Informasi dan Pendidikan
Volume 19, No. 2, September 2026
https://doi.org/10.24036/jtip.v19i2.1088

P.ISSN: 2086 – 4981
E.ISSN: 2620 – 6390
tip.ppj.unp.ac.id

The Impact of Using ChatGPT on Enhancing Students' Programming Ability: An Experimental Study on Prompt Engineering Techniques

Muhammad Hifdzi Adini[1*]✉, Harja Santana Purba[1], Nuruddin Wiranda[1], Muchammad Ardhabilly[1], Rabiatul Adawiyah[1]
[1]Computer Education, Universitas Lambung Mangkurat, Banjarmasin, Indonesia
**Corresponding Author: hifdzi.adini@ulm.ac.id*

Article Information

Article history:

No. 1088
Rec. December 15, 2025
Rev. June 29, 2026
Acc. July 01, 2026
Pub. July 07, 2026
Page. 1664 – 1673

Keywords:

- *ChatGPT*
- *Prompt Engineering Techniques*
- *Programming Ability*
- *Code-to-Code Prompting*
- *Text-to-Code Prompting*

ABSTRACT

The rapid development of Artificial Intelligence (AI) has created new opportunities for supporting programming learning, particularly through the use of ChatGPT and prompt engineering techniques. This study aims to examine the impact of ChatGPT-assisted learning on students' programming ability and to compare the effectiveness of two prompting techniques, namely Text-to-Code and Code-to-Code. A quantitative approach with a pretest-posttest two-group experimental design was employed. The participants consisted of 35 Computer Education students, divided into a Text-to-Code group (n = 18) and a Code-to-Code group (n = 17). Data were collected through programming skill tests and a student perception questionnaire. The results showed that both prompting techniques improved students' programming ability at a medium level. The Text-to-Code group obtained a mean N-Gain score of 0.424, while the Code-to-Code group achieved a higher mean N-Gain score of 0.507. The statistical test indicated a significant difference between the two groups (p = 0.035), suggesting that the Code-to-Code technique was more effective in enhancing programming ability. In addition, students in both groups showed positive perceptions toward the use of ChatGPT, with aggregate mean scores of 4.00 for Text-to-Code and 3.94 for Code-to-Code. These findings indicate that ChatGPT can support programming learning effectively, particularly when students are encouraged to write and analyze code before using AI assistance.

How to Cite:

Adini, M., H., & et al (2026). The Impact of Using ChatGPT on Enhancing Students' Programming Ability: An Experimental Study on Prompt Engineering Techniques. Jurnal Teknologi Informasi Dan Pendidikan, 19(2), 1664-1673. https://doi.org/10.24036/jtip.v19i2.1088

This open-access article is distributed under the Creative Commons Attribution-ShareAlike 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. ©2023 by Jurnal Teknologi Informasi dan Pendidikan.

1. INTRODUCTION

The rapid advancement of technology and Information Technology (IT) has significantly impacted various fields, including Education [1]. Concurrently with this development, Artificial Intelligence (AI) is increasingly utilized in the educational sector, particularly in supporting programming learning. Given the rapid progress of AI technology, the optimization of large language models, such as Chat Generative Pre-trained Transformer (ChatGPT), for specific tasks like computer programming remains an engaging challenge to explore [2]. One of the major challenges students face in learning programming is comprehending algorithmic concepts and programming language syntax.

This challenge can lead to difficulties in solving programming problems and creating efficient code [3]. ChatGPT, as an AI-based language model, has the potential to be an effective tool for enhancing students' programming skills. However, the effectiveness of using ChatGPT depends heavily on the prompting technique employed. A prompt is a brief sentence or paragraph used to initiate a task or text that is to be completed by a language model using AI technology [4]. Effective prompt engineering is crucial to maximize the capabilities of AI tools like ChatGPT within an educational context [5]. Prompt engineering involves the design, refinement, and optimization of inputs (prompts) to effectively communicate the user's intent to the language model [6]. This enables the model to generate relevant, coherent, and high-quality responses [7], [8]. Therefore, this study focuses on two main techniques in prompt engineering, namely code-to-code and text-to-code, to evaluate their impact on the enhancement of students' programming skills.

The code-to-code technique involves providing ChatGPT with existing code snippets for modification, optimization, or analysis. This approach leverages the model's understanding of code structure and syntax to refine the given code [9]. Conversely, the text-to-code technique involves describing the desired program functionality or behavior in natural language, allowing ChatGPT to generate the corresponding code. Both techniques can be tailored to the model's capabilities, potentially mitigating misuse by students [10]. Thus, the code-to-code and text-to-code prompting techniques are critical in determining the effectiveness of ChatGPT as a learning aid.

Previous studies have discussed the use of ChatGPT and prompt engineering strategies in computer programming tasks [2], [3], as well as students' perceptions of ChatGPT in programming learning [4], [9]. However, empirical evidence comparing the effectiveness of specific prompting techniques on students' measurable programming skill improvement remains limited. In particular, few studies have directly compared text-to-code and code-to-code prompting techniques using an experimental pretest-posttest design while also considering students' perceptions of the learning experience. This gap is important because students may perceive an AI-assisted learning technique positively, but such perception does not always correspond to actual improvement in programming ability.

Based on this gap, this study aims to analyze the impact of ChatGPT-assisted learning on students' programming skills and to compare the effectiveness of two prompting techniques: text-to-code and code-to-code. Specifically, this study examines the difference in students' programming skill improvement, measured using the Normalized Gain Score, and analyzes students' perceptions of both prompting techniques in terms of usability, attractiveness, and usefulness. The findings are expected to provide practical insight into how ChatGPT can be integrated into programming learning through appropriate prompting strategies.

2. RESEARCH METHOD

2.1. Research Design

This study employs a quantitative approach with a True Experimental Design, specifically the Pretest-Posttest Two-Group Design. This design was selected because it allows the researcher to compare the outcomes of the intervention (the use of ChatGPT prompting techniques) between two groups that have been randomly assigned. The comparison is focused on two main aspects:

1) Objective Effectiveness: Measured by the difference in skill improvement using the Normalized Gain Score between the two groups.
2) Subjective Effectiveness: Descriptively analyzed based on students' perception scores toward the two prompting techniques.

Table 1. Pretest-Posttest Two-Group Design

Group	Pre-test	Treatment (Intervention)	Post-test	Perception Questionnaire
Experimental Group 1	O_1	X_{T2C} (Text-to-Code)	O_2	P_1
Experimental Group 2	O_3	X_{C2C} (Code-to-Code)	O_4	P_2

2.2. Population and Sample

The population of this research comprises all students in the Computer Education program who are currently enrolled in the Basic Programming course during the first semester. The research sample was obtained using a purposive sampling technique based on specific criteria. The criteria included students who were actively enrolled in the Basic Programming course, had similar initial exposure to programming materials, were willing to participate in the research activities, and had not extensively used ChatGPT for programming tasks prior to the intervention.

After the sample was determined, students were assigned to the treatment groups using a simple random assignment technique. This procedure was carried out to ensure that each participant had an equal opportunity to be placed in either the Text-to-Code or Code-to-Code group, thereby reducing potential selection bias between groups. The Text-to-Code group consisted of 18 students who received treatment using ChatGPT with text-based description instructions. Meanwhile, the Code-to-Code group consisted of 17 students who received treatment using ChatGPT with instructions based on code snippets or code analysis.

P.ISSN: 2086 – 4981
E.ISSN: 2620 – 6390
tip.ppj.unp.ac.id

2.3. Research Variables

This study involves two main types of variables:

1) Independent Variable: ChatGPT Prompting Technique, which consists of two levels of treatment: Text-to-Code and Code-to-Code.
2) Dependent Variables: The study measures two key outcomes:
   - Programming Skill: Measured via the Normalized Gain (N-Gain) Score derived from the Pre-test and Post-test results.
   - Student Perception: Measured by the aggregate score from the questionnaire, which covers the dimensions of Usability, Attractiveness, and Usefulness.

2.4. Research Instruments and Data Analysis Techniques

Two types of instruments were utilized for data collection: the programming skill test and the perception questionnaire. The programming skill test was administered as a pre-test and post-test to measure students' programming ability before and after the intervention. The test was used to assess students' ability to solve basic programming problems based on the learning materials provided during the experimental activities. The pre-test was conducted before the treatment, while the post-test was conducted after students received the ChatGPT-assisted learning treatment using the assigned prompting technique.

The perception questionnaire was used to measure students' responses toward the implemented prompting techniques. The questionnaire consisted of 12 items covering three aspects: usability, attractiveness, and usefulness. Each item was measured using a five-point Likert scale. The reliability of the questionnaire was examined using Cronbach's Alpha to determine the internal consistency of the items.

Data analysis was conducted in two main stages. First, the improvement in students' programming skills was analyzed using the Normalized Gain Score (N-Gain). The N-Gain score was calculated from the difference between the pre-test and post-test scores and was used to determine the level of students' programming skill improvement after the intervention. The N-Gain score was calculated using the following formula:

$$g = (\text{post-test score} - \text{pre-test score}) / (\text{maximum score} - \text{pre-test score})$$

The N-Gain results were interpreted into three categories: low improvement ($g \leq 0.3$), medium improvement ($0.3 < g \leq 0.7$), and high improvement ($g > 0.7$). The mean pre-test score, mean post-test score, mean N-Gain score, and improvement category were then presented descriptively for each group.

Second, students' perception data were analyzed using descriptive statistics by calculating the mean and standard deviation for each questionnaire item and the aggregate perception score for each group. The Cronbach's Alpha coefficient was also reported to

show the reliability of the perception questionnaire. To examine whether there was a statistically significant difference in programming skill improvement between the Text-to-Code and Code-to-Code groups, the Mann-Whitney U Test was applied to the N-Gain scores. This non-parametric test was selected because the study involved two independent groups with relatively small sample sizes and did not rely on the assumption of normally distributed data. The statistical decision was based on the significance value (p-value) at the 0.05 level. Meanwhile, students' perception data were analyzed descriptively using the mean and standard deviation for each questionnaire item and the aggregate perception score for each group.

3. RESULTS AND DISCUSSION

This section presents the findings derived from the experimental analysis designed to compare the effectiveness of the Text-to-Code and Code-to-Code prompting techniques using ChatGPT on students' programming skills. The results are divided into two main sections: objective effectiveness, measured by the Normalized Gain (N-Gain) score, and subjective effectiveness, assessed through the student perception questionnaire. The analysis focuses on students' programming skill improvement and their perceptions of the usability, attractiveness, and usefulness of each prompting technique.

3.1. Analysis of Skill Enhancement (Normalized Gain Score)

To objectively measure the effectiveness of the intervention, the Normalized Gain (N-Gain) score was calculated. The N-Gain score was used to determine the extent to which students' programming skills improved from the pre-test to the post-test after receiving the ChatGPT-assisted learning treatment.

Table 2. Descriptive Results of Normalized Gain Score

Group	N	Mean Pre-test	Mean Post-test	Mean N-Gain Score (g)	Improvement Category
Text-to-Code	18	51.11	71.89	0.424	Medium
Code-to-Code	17	51.41	76.06	0.507	Medium
N-Gain Difference				0.083	

The descriptive results show that both prompting techniques improved students' programming skills. The Text-to-Code group obtained a mean N-Gain score of 0.424, while the Code-to-Code group obtained a higher mean N-Gain score of 0.507. Based on the N-Gain interpretation criteria, both groups were categorized as having medium improvement, with the medium category defined as $0.3 < g \leq 0.7$ [11]. Although both techniques resulted in moderate improvement, the Code-to-Code technique showed a higher average improvement than the Text-to-Code technique, with an N-Gain difference of 0.083.

To determine whether the difference in programming skill improvement between the two groups was statistically significant, a comparative test was conducted on the N-Gain

P.ISSN: 2086 – 4981
E.ISSN: 2620 – 6390
tip.ppj.unp.ac.id

scores using the Mann-Whitney U Test. This test was selected because the study involved two independent groups with relatively small sample sizes and did not rely on the assumption of normally distributed data [12]. The statistical decision was based on the significance value (p-value) at the 0.05 level.

Table 3. Mann-Whitney Test Result of N-Gain Scores

Variable	Group Comparison	Sig. (p-value)	Interpretation
N-Gain Score	Text-to-Code vs Code-to-Code	0.035	Significant

The test result showed a significance value of p = 0.035, which is lower than the significance level of 0.05. This result indicates that there was a statistically significant difference in programming skill improvement between the Text-to-Code and Code-to-Code groups. Therefore, the Code-to-Code prompting technique can be considered more effective than the Text-to-Code technique in enhancing students' programming skills.

This finding suggests that the Code-to-Code technique provides a stronger learning effect because students are required to write, read, analyze, or revise code before receiving assistance from ChatGPT. In this technique, ChatGPT functions not as the primary generator of the solution, but as a tool for feedback, correction, debugging, and optimization. This process encourages students to engage more actively with programming logic and code structure. Previous studies also emphasize that ChatGPT-assisted programming learning can support code explanation, debugging assistance, and personalized feedback when its use is guided by appropriate instructional strategies [13], [14].

Thus, the results support the view that ChatGPT can be beneficial in programming learning when its use is structured to encourage students' active cognitive involvement. The Code-to-Code approach appears to promote deeper engagement because students must first attempt to formulate a coding solution before using AI assistance. This finding has important implications for programming instruction, suggesting that AI-assisted learning should be designed not merely to provide direct answers, but to guide students in analyzing, correcting, and improving their own programming solutions [15], [16], [17], [18].

3.2. Analysis of Respondent Perception

In addition to measuring students' programming skill improvement, this study also analyzed students' perceptions of the Text-to-Code and Code-to-Code prompting techniques. The perception questionnaire consisted of 12 items covering three aspects: usability, attractiveness, and usefulness. Before analyzing the perception scores, a reliability test was conducted using Cronbach's Alpha to examine the internal consistency of the questionnaire items.

Table 4. Reliability Test

Group	Items (N)	Cronbach's Alpha (α)	Description
Text-to-Code	12	0.916	Excellent
Code-to-Code	12	0.909	Excellent

The reliability test results show that the Cronbach's Alpha values for both groups were above 0.90. The Text-to-Code group obtained an alpha value of 0.916, while the Code-to-Code group obtained an alpha value of 0.909. These results indicate that the questionnaire had excellent internal consistency and was reliable for measuring students' perceptions of the implemented prompting techniques [19].

Table 5 presents the comparison of mean and standard deviation scores for each perception item between the Text-to-Code and Code-to-Code groups.

Table 5. Comparison of Mean and Standard Deviation Scores for Perception Items

No.	Perception Item	Text-to-Code (Mean ± SD)	Code-to-Code (Mean ± SD)
1	The prompting technique was easy for me to understand.	4.28 ± 0.67	4.29 ± 0.69
2	I was able to write prompts that produced code according to my needs.	3.89 ± 0.83	3.82 ± 0.88
3	The code generated by ChatGPT could be used directly.	3.94 ± 0.73	3.88 ± 0.60
4	I found it easy to learn from the code changes generated by ChatGPT.	3.89 ± 0.76	3.76 ± 0.73
5	This technique helped me understand program structure.	4.06 ± 0.80	4.06 ± 0.75
6	I understood program logic flow better.	3.94 ± 0.87	3.82 ± 0.95
7	This technique improved my ability to write code independently.	3.78 ± 0.94	3.82 ± 0.95
8	ChatGPT helped me identify programming errors.	3.72 ± 0.75	3.88 ± 0.60
9	Programming learning became more interesting.	4.33 ± 0.69	4.29 ± 0.69
10	I was encouraged to experiment further.	4.06 ± 0.73	3.94 ± 0.83
11	This technique helped me complete programming tasks more quickly.	4.22 ± 0.73	4.06 ± 0.75
12	I perceived an improvement in my algorithmic ability.	3.83 ± 0.79	3.71 ± 0.92
Aggregate Mean of Overall Perception		4.00 ± 0.52	3.94 ± 0.54

The results show that students in both groups had positive perceptions of the implemented prompting techniques. The aggregate perception score of the Text-to-Code group was 4.00, while the Code-to-Code group obtained an aggregate score of 3.94. These scores indicate that both techniques were perceived positively by students in terms of usability, attractiveness, and usefulness.

The highest-rated item in both groups was related to the statement that programming learning became more interesting. The Text-to-Code group obtained a mean score of 4.33 for this item, while the Code-to-Code group obtained a mean score of 4.29. This finding suggests that the use of ChatGPT in programming learning may increase students' interest and motivation, regardless of the prompting technique used. Similar findings have

P.ISSN: 2086 – 4981
E.ISSN: 2620 – 6390
tip.ppj.unp.ac.id

been reported in studies showing that students generally perceive ChatGPT as useful, accessible, and supportive in programming learning activities [13], [14].

Several items also show that students perceived ChatGPT as useful for supporting programming tasks. For example, the Text-to-Code group gave a relatively high score to the statement that the technique helped them complete programming tasks more quickly, with a mean score of 4.22. Meanwhile, the Code-to-Code group gave relatively high scores to the items related to ease of understanding the technique and increased attractiveness of programming learning, with mean scores of 4.29 for both items.

Although the Text-to-Code group obtained a slightly higher aggregate perception score than the Code-to-Code group, the difference was small. Therefore, based on the descriptive results, both prompting techniques can be considered positively received by students. However, this descriptive perception result should be interpreted carefully because a positive perception does not necessarily indicate a higher improvement in programming skills. As shown in the N-Gain analysis, the Code-to-Code technique produced a higher improvement in programming ability, even though students' perception scores for the two techniques were relatively similar. Previous studies have also noted that the use of generative AI may improve efficiency and learning experience, but it can also create risks of over-reliance and reduced cognitive engagement if not integrated through structured learning activities [15], [18], [20].

These findings imply that students may value ChatGPT-assisted learning because it makes programming activities more engaging, accessible, and efficient. Nevertheless, the pedagogical effectiveness of a prompting technique should not be evaluated only from students' perceptions. In programming learning, the design of AI-assisted activities needs to ensure that students remain cognitively engaged in understanding logic, analyzing code, identifying errors, and constructing solutions independently.

4. CONCLUSION

Based on the results of the study, it can be concluded that both ChatGPT-based prompting techniques, namely Text-to-Code and Code-to-Code, contributed to the improvement of students' programming skills. The Text-to-Code group obtained a mean N-Gain score of 0.424, while the Code-to-Code group obtained a higher mean N-Gain score of 0.507. Both scores were categorized as medium improvement. The comparison test showed a significance value of $p = 0.035$, indicating that there was a statistically significant difference in programming skill improvement between the two groups. These results suggest that the Code-to-Code prompting technique was more effective than the Text-to-Code technique in enhancing students' programming skills.

The findings also show that students had positive perceptions of both prompting techniques. The aggregate perception score of the Text-to-Code group was 4.00, while the Code-to-Code group obtained an aggregate score of 3.94. The reliability of the perception

questionnaire was also excellent, with Cronbach's Alpha values of 0.916 for the Text-to-Code group and 0.909 for the Code-to-Code group. These results indicate that students perceived ChatGPT-assisted programming learning as useful, interesting, and supportive of their learning activities. However, the descriptive perception results also show that positive student perception does not always correspond directly to higher programming skill improvement.

From a practical perspective, the results of this study suggest that ChatGPT should be integrated into programming learning through structured prompting activities. The Code-to-Code technique is recommended because it encourages students to write, analyze, revise, and reflect on code before receiving assistance from ChatGPT. In this way, ChatGPT can function as a learning support tool for feedback, debugging, and refinement rather than merely as a tool for generating direct answers. Therefore, programming instructors are encouraged to design AI-assisted learning activities that promote students' cognitive engagement and independent problem-solving skills.

This study has several limitations. The number of participants was relatively small and involved students from one study program, so the findings should be generalized with caution. In addition, this study focused only on two prompting techniques and measured programming skill improvement through pre-test, post-test, N-Gain, and student perception data. Future research is recommended to involve a larger sample, include more varied programming topics, compare additional prompting strategies, and examine long-term effects of ChatGPT-assisted learning on students' programming competence.

REFERENCES

[1] R. A. Sukmawati, M. H. Adini, M. Pramita, and A. Rizqan, "Implementasi Gamifikasi Pada Pengembangan Multimedia Pembelajaran Interaktif Dengan Metode *Drill and Practice*," *EDU-MAT: Jurnal Pendidikan Matematika*, vol. 9, no. 2, p. 163, Nov. 2021, doi: 10.20527/edumat.v9i2.11728.

[2] N. M. Mnguni, N. Nkomo, K. Maguraushe, and M. B. Mutanga, "An Experimental Study of The Efficacy of Prompting Strategies In Guiding ChatGPT for A Computer Programming Task," *Journal of Information Systems and Informatics*, vol. 6, no. 3, pp. 1346–1359, Sep. 2024, doi: 10.51519/journalisi.v6i3.783.

[3] T. Wang, N. Zhou, and Z. Chen, "Enhancing Computer Programming Education with LLMs: A Study on Effective Prompt Engineering for Python Code Generation," Jul. 2024, [Online]. Available: http://arxiv.org/abs/2407.05437

[4] R. Yilmaz and F. G. Karaoglan Yilmaz, "Augmented intelligence in programming learning: Examining student views on the use of ChatGPT for programming learning," *Computers in Human Behavior: Artificial Humans*, vol. 1, no. 2, p. 100005, Aug. 2023, doi: 10.1016/j.chbah.2023.100005.

[5] D. Lee and E. Palmer, "Prompt engineering in higher education: a systematic review to help inform curricula," *International Journal of Educational Technology in Higher Education*, vol. 22, no. 1, Dec. 2025, doi: 10.1186/s41239-025-00503-7.

[6] S. Ekin, "Prompt Engineering For ChatGPT: A Quick Guide To Techniques, Tips, And Best Practices," May 04, 2023. doi: 10.36227/techrxiv.22683919.v2.

[7] M. Kim and L. Adlof, "Adapting to the Future: ChatGPT as a Means for Supporting Constructivist Learning Environments," *TechTrends*, vol. 68, no. 1, pp. 37–46, 2024, doi: 10.1007/s11528-023-00899-x.

P.ISSN: 2086 – 4981
E.ISSN: 2620 – 6390
tip.ppj.unp.ac.id

[8] P. P. Pawar, K. B. Salve, and R. R. Patil, "'Impact of ChatGPT on Student's Education: A Comprehensive analysis of positive and negative effects,'" *Journal of Advanced Zoology*, vol. 44, no. S8, 2023, doi: 10.53555/jaz.v44iS8.3504.
[9] K. Aruleba, I. T. Sanusi, G. Obaido, and B. Ogbuokiri, "Integrating ChatGPT in a Computer Science Course: Students Perceptions and Suggestions," Dec. 2023, doi: 10.48550/arXiv.2402.01640.
[10] B. Cowan, Y. Watanobe, and A. Shirafuji, "Enhancing Programming Learning with LLMs: Prompt Engineering and Flipped Interaction," in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Oct. 2023, pp. 10–16. doi: 10.1145/3634814.3634816.
[11] R. R. Hake, "Interactive-engagement versus traditional methods: A six-thousand-student survey of mechanics test data for introductory physics courses," *Am. J. Phys.*, vol. 66, no. 1, pp. 64–74, Jan. 1998, doi: 10.1119/1.18809.
[12] H. B. Mann and D. R. Whitney, "On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other," *The Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50–60, Mar. 1947, doi: 10.1214/AOMS/1177730491.
[13] D. Sun, A. Boudouaia, C. Zhu, and Y. Li, "Would ChatGPT-facilitated programming mode impact college students' programming behaviors, performances, and perceptions? An empirical study," *International Journal of Educational Technology in Higher Education*, vol. 21, no. 1, Dec. 2024, doi: 10.1186/s41239-024-00446-5.
[14] H. Güner and E. Er, "AI in the classroom: Exploring students' interaction with ChatGPT in programming learning," *Education and Information Technologies 2025 30:9*, vol. 30, no. 9, pp. 12681–12707, Jan. 2025, doi: 10.1007/S10639-025-13337-7.
[15] E. Kasneci *et al.*, "ChatGPT for good? On opportunities and challenges of large language models for education," *Learn. Individ. Differ.*, vol. 103, p. 102274, Apr. 2023, doi: 10.1016/J.LINDIF.2023.102274.
[16] C. K. Lo, "What Is the Impact of ChatGPT on Education? A Rapid Review of the Literature," *Educ. Sci. (Basel).*, vol. 13, no. 4, p. 410, Apr. 2023, doi: 10.3390/EDUCSCI13040410/S1.
[17] S. Li, J. Liu, and Q. Dong, "Generative artificial intelligence-supported programming education: Effects on learning performance, self-efficacy and processes," *Australasian Journal of Educational Technology*, vol. 41, no. 3, pp. 1–25, May 2025, doi: 10.14742/AJET.9932.
[18] T. C. Yang, Y. C. Hsu, and J. Y. Wu, "The effectiveness of ChatGPT in assisting high school students in programming learning: evidence from a quasi-experimental research," *Interactive Learning Environments*, vol. 33, no. 6, pp. 3726–3743, Jul. 2025, doi: 10.1080/10494820.2025.2450659.
[19] M. Tavakol and R. Dennick, "Making sense of Cronbach's alpha," *Int. J. Med. Educ.*, vol. 2, pp. 53–55, Jun. 2011, doi: 10.5116/IJME.4DFB.8DFD/RM.
[20] D. Stoyanova, S. Stoyanova-Petrova, S. Shotarova, S. Lyubomirov, and N. Mileva, "ChatGPT in Programming Education: An Empirical Study on Its Impact on Student Performance, Creativity, and Teamwork," *Education Sciences 2026, Vol. 16, Page 19*, vol. 16, no. 1, p. 19, Dec. 2025, doi: 10.3390/EDUCSCI16010019.