ASCILITE 2024

Navigating the Terrain:

Emerging Frontiers in Learning Spaces, Pedagogies, and Technologies

Challenges and opportunities in using ChatGPT as a team member to promote code review education and self-regulated learning

Paula G. de Barba, Chakkrit (Kla) Tantithamthavorn
Monash University, Australia

In this study, we examine the use of ChatGPT as a team member to enhance code review education and promote self-regulated learning among programmers of varying experience levels. A case-study approach was used to answer three research questions about code quality, perceived benefits, and challenges. Participants reported diverse perspectives on the quality and utility of ChatGPT-generated code, with novice programmers gaining a deeper understanding of code interactions and experienced programmers refining their code through iterative prompt optimisation. Despite the benefits, challenges emerged, such as concerns about the potential negative impact of generative AI on learning experiences and the difficulties posed by foundational knowledge gaps in effectively interacting with ChatGPT. The findings suggest that ChatGPT is a promising tool for supporting code review education, especially for novice programmers, and for fostering self-regulated learning strategies like elaboration and metacognition. Future research should validate these results with larger sample sizes and address the identified challenges to better integrate generative AI tools in educational contexts.

Keywords: generative artificial intelligence, code review, self-regulated learning, computer science, higher education, case study.

Introduction

An important part of software engineers' jobs includes being able to review their peers' codes to improve code quality. Developing code-reviewing skills, which encompasses both technical and interpersonal factors, is currently part of most software engineering curricula in higher education. Many successful practices to promote the learning of these skills have emerged over the last two decades, such as peer code review education (Indriasari et al., 2020). Promoting code review education involves several pedagogical strategies, with peer code review becoming a common approach in higher education over the past years (Indriasari et al., 2020). Peer code review encompasses students submitting their work, which is then reviewed by other students who assess its code quality. This process usually happens anonymously in small groups, with students using a marking guide or checklist to assist them in providing effective feedback. Generative artificial intelligence (AI) is a promising tool to be used for this purpose, as it can act as a team member (Mollick & Mollick, 2023) to generate the code for students to review. However, little is known about the ChatGPT-generated code quality, the benefits and the challenges of using a generative artificial intelligence tool for this educational purpose.

Underpinning many students' programming skills, including code-reviewing, is their ability to regulate their own learning (Silva et al., 2024), which is broadly defined as self-regulated learning. Self-regulated learning relates to how students plan, monitor and evaluate their learning (for review, see Panadero, 2017). This requires them to use specific sets of learning strategies, such as connecting new information to prior knowledge (i.e., elaboration), thinking about their thinking or learning (i.e., metacognition), and critically evaluating new knowledge (i.e., critical thinking; Pintrich et al., 1991). Such skills have also been flagged as crucial for students to learn how to navigate the use of generative artificial intelligence in educational settings (Lodge et al., 2023).

In this project, we investigated students' perceptions of using generative artificial intelligence, particularly ChatGPT (https://www.chat.openai.com), during a code review assessment task from a case study approach.

We hypothesise that by using ChatGPT as a team member, students may enhance both their code-reviewing and self-regulated learning skills. The significance of this research lies in its potential to uncover some of the challenges and benefits of using ChatGPT as a team member in assessment tasks. The following research questions were investigated:

1. What code quality issues are found in the ChatGPT-generated code and how can they be addressed?
2. What are the benefits of using ChatGPT to support code review and self-regulated learning?
3. What are the challenges of using ChatGPT to support code review learning?

Literature review

ChatGPT presents significant potential to be used as a team member for code review education due to its ability to generate useful yet imperfect code. These imperfections, such as issues with code maintainability and implementation (Liu et al., 2023), mirror the types of errors students need to learn to identify in real-world coding projects. The nature of these errors varies depending on the complexity of the task, the programming language used, and regardless of the code being functional or not (i.e., even functional codes can have these types of errors; Liu et al., 2023), making ChatGPT suitable for different learning scenarios. Furthermore, research has shown that these errors can be corrected through prompt adjustment and the provision of precise feedback (Liu et al., 2023).

Recent studies also highlight the benefits of using ChatGPT as a team member in software development projects to generate code. Lin and colleagues (2024) found that students who used ChatGPT as a team member in a software development project assessment valued its ability to simulate real-time team dynamics and generate code snippets, enhancing their motivation to work on the project. Similarly, Waseem and colleagues (2024) found that using ChatGPT as a team member had a positive impact on their learning, while appreciating the challenges faced, such as ChatGPT's constraints in dealing with more complex tasks, as an important and constructive part of the learning experience.

However, there has been limited investigation into the use of ChatGPT-generated code as part of and as a team member specifically for code review education. Code review education requires students to provide effective feedback to their peers, usually their own team members. Similar peer interactions in assessments have been found to have negative impact of students' low levels of social connectedness and feelings of safety on their engagement with the task and their ability in giving effective feedback (Panadero et al., 2023). From this perspective, ChatGPT has the potential to provide a safe space, similar to the ones provided by simulations in other domains (e.g., teacher training; Walker et al., 2021), where students can focus on the technical aspect of learning without the complexities of interpersonal dynamics. Technical code review skills aim to identify broader programming errors (or bugs) than those identified by automated testing, ensuring code maintainability, and improving its comprehension and implementation (Chong et al., 2021). Once students have developed these technical review skills, additional layers, such as interpersonal factors, could be introduced to enhance their overall competency in code review. This is aligned with Hattie and Timperley's (2007) phased approach to providing feedback to students; that is, first focusing on providing task-level feedback (i.e., technical code review skill) to then providing process-level feedback (i.e., interpersonal skills).

Despite the potential benefits of using ChatGPT to facilitate code review education, its integration as a team member is not without challenges. In the study conducted by Lin and colleagues (2024) which investigated the use of ChatGPT as a team member in software development projects, some students reported negative perceptions about over-reliance on ChatGPT, hindering their critical thinking and problem-solving skills, and ChatGPT's limitations in handling ambiguity or vague requirements. Additionally, even though we assume there will be reduced or no interpersonal factors impacting their interaction with a generative AI chatbot, previous research suggests that some individuals may perceive a chatbot as having its own mind while some may not (Lee et al., 2020). This difference has been found to impact people's interaction with chatbots and overall satisfaction (e.g., Yam et al., 2021). Thus, while ChatGPT presents a promising tool for enhancing technical skills and motivation in code review education, it is crucial to better understand challenges related to hindering students' critical thinking skills, which is a key component of self-regulated learning.

Self-regulated learning is about students taking agency of their learning and bringing their own experiences and awareness to their learning experience (Lodge et al., 2023). In this study, we focus on investigating students' use of three learning strategies, which have been found to be relevant to their interaction with ChatGPT during a code review task (Silva et al., 2024). The first learning strategy is elaboration, which is related to connecting new information to prior knowledge (Pintrich et al., 1991). While reviewing code, students are expected to compare the code being reviewed with their prior knowledge about code quality and programming techniques. The second learning strategy is metacognition, which is when students distance themselves from their thinking to examine it (Pintrich et al., 1991). This is important during code review as students must recognise when they are missing specific knowledge to review a code. Once this is recognised, they may be able to pursue that knowledge to then review the code. The third learning strategy is critical thinking, which is about critically evaluating new knowledge (Pintrich et al., 1991). For code review, this means that students will check for evidence and scrutinise any solutions the ChatGPT presents to them, instead of simply accepting them as valid or correct.

Method

A 6-week Software Engineering unit, part of a fully online postgraduate course in Computer Science in Australia, was undergoing assessment restructuring. The authors identified this as a good opportunity to implement the use of ChatGPT as a team member and investigate students' perceptions of that learning experience. The reason to include ChatGPT as part of the assessment was two-fold: (a) reduce workload for the final assessment related to coding so students could focus on the project management aspects of the task to align with the unit's learning outcomes, including code review rather than on coding itself, and (b) provide an opportunity for students to experience using ChatGPT as an important skill in the computer science field.

At the end of the teaching period, all 59 students were invited to complete a Google Forms survey to investigate their perceived challenges and benefits of using ChatGPT as a team member in one of their assessment tasks (Ethics application #38742). We didn't provide any incentives for participants. The summative assessment that included the use of ChatGPT was their final group assessment, worth 50% of their final grade, encompassing the development and implementation of a particular software. The first step in completing this assessment was interacting with ChatGPT, prompting it to create the initial code for the software. In the second step, students then reviewed the code for the following assessment steps (i.e., implementation). Students had access to instructional material on using ChatGPT as a team member, such as a YouTube video (Daschner, 2022), and instructions on creating a code review checklist based on previous research (Chong et al., 2021). Code review was explicitly taught in one of their unit modules.

The mixed methods survey included open and closed questions. Firstly, we asked participants their age (range selection), their programming experience (scale from 1 '*very inexperienced*' to 10 '*very experienced*'; Feigenspan et al., 2012), and the ChatGPT version used (*3.5 free version*, *3.5 plus*, *4*, and *I don't know*). In relation to their experience using ChatGPT in their assessment, we asked them about:

- preference in using ChatGPT in the future for coding, with three response options (*generate the first draft of code using ChatGPT or similar generative AI*, *program by myself from scratch*, and *other*);
- satisfaction with the ChatGPT-generated code, with a scale ranging from 1 (*Very unsatisfied*) to 10 (*Very satisfied*);
- self-regulated learning strategies they agree or disagree to have used when engaging with ChatGPT-generated code in their assessment, with a scale ranging from 1 (*Totally disagree*) to 7 (*Totally agree*; statements presented in the paragraph below);
- whether they believed using ChatGPT was useful in promoting students' learning during code review education, with a scale ranging from 1 (*Not at all*) to 10 (*Very much*);

- what were the quality issues they found in ChatGPT-generated code and how they can be addressed, with an open-text response field;
- what challenges did they find in using ChatGPT to support their code review learning experience, with an open-text response field.

The three self-regulated learning strategies investigated included one statement on elaboration (*When reviewing the code, I tried to relate it to software engineering content I already knew*), one statement on metacognition (*When reviewing the code, I tried to determine which software engineering concepts I didn't understand well*) and fours statements on critical thinking (*I found myself questioning ChatGPT to decide if I found the code generated convincing; When reviewing the code, I tried to decide if there was good supporting evidence for the way it was presented; I treated the code as a starting point and tried to develop my own ideas about it; When reviewing the code, I thought about possible programming/coding alternatives*). These items were selected and adapted from Meijs and colleagues' (2019) adult distance education subscales complex cognitive strategy use and academic thinking, based on their suitability for the current learning context (i.e., generate code using ChatGPT).

We used a case study approach to analyse the data, providing an in-depth examination of each student's perceptions of using ChatGPT as a team member. This approach was particularly effective given our small sample size (n=4), allowing us to examine their detailed insights. We used Google Sheets to analise both the quantitative and the qualitative data (thematic analysis; Braun & Clarke, 2006) to comprehensively understand the students' experiences and viewpoints.

Results

Four participants with varying prior programming experience answered the survey, with only three fully completing it. This section presents each case in Table 1 and then answers each research question considering data from all cases. For the latter part of this section, we mainly focus on comparing Cases 1 and 2 with Case 3 based on their different programming experience: Cases 1 and 2 reported having some experience in programming, while Case 3 reported being very inexperienced. Case 4, which reported being quite experienced in programming, only answered questions about quality issues in ChatGPT-generated code (RQ1).

Table 1
Individual cases information

	Case 1	Case 2	Case 3	Case4
Age	21 to 30	21 to 30	21 to 30	41 to 50
ChatGPT version used	4	3.5 Plus	3.5 Free	3.5 Free
Programming experience*	4	4	1	8

Note. Programming experience varied from 1 (*Very inexperienced*) to 10 (*Very experienced*).

RQ1: Quality issues in ChatGPT-generated code

The participant with very little experience in programming (Case 3) reported they were very unsatisfied with the quality of the ChatGPT-generated code (1 out of a 10-point scale). They said that there were some quality issues in the ChatGPT-generated code, which they would then ask ChatGPT to correct, but some errors would remain (see quote below, Case 3). Even though the participant acknowledges ChatGPT's limitations, they did not report trying to find an explanation for ChatGPT's error.

> '*If I have* [an] *error, I just post the error to* [C]*hat*[GPT] *and request it to generate a new code that solves that error, but it still got errors* [after that].' (Case 3).

Participants with some programming experience (Cases 1 and 2) were satisfied with the quality of the ChatGPT-generated code (8 and 7 out of a 10-point scale, respectively). They particularly mentioned ChatGPT's limitations in relation to more complex requests involving multiple classes or large-scale projects and its coding style.

> *'Quality issues were minimal until multiple classes were involved, involving complex functions.'* (Case 1)
>
> *'In Java, ChatGPT does not make private variables or class methods to break up the code for the class. It tends to put all the code into the main method. I need to ask it to break up the code into attributes and methods. The code is functional but not readable and hard to maintain initially.* [...] *ChatGPT may also tend to go off track if given a large prompt and subsequent prompts can lead ChatGPT astray.* [...] *ChatGPT also cannot make a large scale project all at once.'* (Case 2)

Extending on the quality of the ChatGPT-generated code, they highlighted the need to work within ChatGPT constraints and created explanations on why the error was happening. These included accepting ChatGPT's limited capacity, as highlighted in the excerpts below:

> '*Given ChatGPT's somewhat limited memory, trying to have ChatGPT do what you need for larger, complex tasks, involving many factors becomes more difficult, and therefore the quality drops*' (Case 1).
>
> '*ChatGPT also cannot make a large scale project all at once, although this may be a reasonable limitation due to the token and context limitations in the back-end of the AI*' (Case 2).

Related to this, they mentioned the importance of prompt engineering for generating better-quality code.

> *'at that point* [when code quality is poor] *it is about understanding prompt engineering and knowing what to tell* [C]*hat*[GPT] *and when to tell it, in order to get the right outcome* [...] *(basically feeding* [ChatGPT] *appropriate acceptance criteria)'* (Case 1).
>
> '*ChatGPT may also tend to go off track if given a large prompt and subsequent prompts can lead ChatGPT astray. This is addressed by making prompts small and concise and focused on fixing specific bugs or asking specific questions about the code.*' (Case 2).

The most experienced participant in programming (Case 4) was satisfied with the quality of the ChatGPT-generated code (7 out of a 10-point scale). They particularly commented on the importance of prompt engineering for generating better-quality code, saying that it '*depend*[s] *on what you are asking and what is the context*' (Case 4).

When asked whether they would use ChatGPT to generate code in future programming tasks, only the experienced participant selected the option to program everything from scratch (Case 4). All other participants said that they would prefer to generate the first draft of the code using ChatGPT and improve it from there.

RQ2: Benefits of using ChatGPT for code review and self-regulated learning

Participants with some programming experience stated that using ChatGPT would slightly (4 out of a 10-point scale; Case 1) and significantly (7 out of a 10-point scale; Case 2) enhance students' learning during code review education. The participant with very little experience in programming (Case 3) reported that using ChatGPT would greatly (10 out of a 10-point scale) enhance students' learning during code review education.

Regarding the benefits of promoting the use of self-regulated learning strategies, all participants agreed (6 or 7 out of a 7-point scale) that using ChatGPT to support their code review learning promoted their use of elaboration, metacognition, and one aspect of critical thinking related to using the ChatGPT-generated code as a starting point. Other aspects of critical thinking had lower ratings (average of 3.44, ranging from 2 to 5 out of a 7-point scale), such as questioning the quality of the code, looking for good supporting evidence and thinking about alternative solutions, meaning that these were not clearly promoted just by using the ChatGPT when code reviewing.

> *'ChatGPT promotes learning as it helps to teach you the bigger picture. You learn and experience how classes and functions interact, beyond the entry-level knowledge.* [...] *I was able to follow the code along and use my understanding of what is required and when to get* [C]*hat*[GPT] *to build what I wanted. In this way, I would say* [C]*hat*[GPT] *is excellent.'* (Case 1)

RQ3: Challenges using ChatGPT to support the code review learning experience

One of the participants with some experience in programming related that even though using ChatGPT did support their learning experience, they expressed concerns that it was hindering their opportunity to learn to code:

> '*Because the code generation was automated, my coding ability barely increased, in the real world, I will need to spend 3 months learning to code without* [C]*hat*[GPT] *before I apply to careers, because you become reliant on* [C]*hat*[GPT]. [...] *you find yourself eyeing over the code* [C]*hat*[GPT] *generates to understand what it's doing, and how it's doing it, which is helpful, but could I myself then write anywhere near that level of code without* [C]*hat*[GPT]*'s help? No*' (Case 1)

The other participant with some programming experience highlighted the lack of sustainability of the ChatGPT-generated code, which is related to some of the quality issues related to ChatGPT's ability to handle large-scale projects.

> *'ChatGPT is mostly a lesson on how to not style code. It is very good for rapid prototyping of software but not maintainable in the long term.'* (Case 2)

The participant with very little programming experience listed their own lack of experience as a challenge to using ChatGPT for code review education.

> *'I guess is my lack of coding experience and knowledge from the start. I did finish a Java programming course, python and [SQL] in these units but I have to be honest that I don't really know much and just got a pass in these units. Even if* [C]*hat*[GPT] *provides the answer, I still need extensive clarification and explanation from the chatbot.'* (Case 3).

Discussion and implications

The current research allowed us to explore the three research questions using a case study approach. Regarding the first research about the quality issues found in the ChatGPT-generated code, participants' responses varied according to their level of programming experience: more experienced programmers were generally satisfied with the ChatGPT-generated code, while the very inexperienced programmer was very unsatisfied. The quality issues were related to ChatGPT's inability to deal with more complex requests and its poor coding style.

An interesting finding related to the quality issues raised was that students with more experience programming went beyond simple quality judgements (i.e., simply good or bad), seeking explanations (e.g., ChatGPT's limited memory) and ways to improve (e.g., using targeted prompts) the quality of the ChatGPT-generated code. This supports previous research that users' perception of a chatbot impacts their interactions (Lee et al., 2020), particularly related to how it can guide their use of critical thinking skills (Hyytinen et al., 2014). That is, depending on how an individual conceptualises the source of knowledge, it could either engage with them thoroughly or superficially (Hyytinen et al., 2014). If individuals believe that it is possible for a source of knowledge to improve and that not all knowledge is solely constructed by human minds, they use critical thinking as a tool for enhancing understanding (thorough critical thinking). However, if they focus on isolated details and accept knowledge as either true or false, without much questioning, they use critical thinking as a tool for determining truth or falsehood (superficial critical thinking). This suggests that the effectiveness of using generative AI as an educational tool may depend on users' trust and belief in the tool's potential for future enhancements and knowledge expansion.

Although students identified errors in the ChatGPT-generated code, most of them still selected that they would prefer to use ChatGPT-generated code to start new projects rather than programming everything from scratch - including the very inexperienced programmer who was very unsatisfied with the ChatGPT-generated code. The only participant who preferred to program everything from scratch was the most experienced programmer. This may be related to (a) their preference as this has been the way they have always been programming, (b) it would be hard for the less inexperienced programmers to actually code as fast as ChatGPT, and/or (c) it could mean that having more experience programming allows them to identify more errors, therefore deeming fixing the initial code more effortful than starting from scratch. Options (b) and (c) require closer examination as they could reflect a gap in programming skills or knowledge that needs to be addressed when using ChatGPT for programming tasks.

The perceived benefits of using ChatGPT for code review education varied according to participants' levels of programming experience. Participants with lower programming experience found ChatGPT to be highly beneficial for code review education. This suggests that ChatGPT may be particularly beneficial for novice programmers, helping them to understand interactions within the code (e.g., '*You learn and experience how classes and functions interact, beyond the entry-level knowledge.'* Case 1) and providing a scaffolded learning experience if they required further explanation (e.g. '*Even if* [C]*hat*[GPT] *provides the answer, I still need extensive clarification and explanation from the chatbot'.* Case 3). One student noted that ChatGPT helps "*teach you the bigger picture*" (Case 1) by illustrating how different parts of the code interact. Perhaps focusing solely on the code review aspect of programming (i.e., understanding and interpreting the code rather than producing a code from scratch), allowed them to apply and interact with their knowledge differently.

This is supported by the findings that these participants also reported perceived benefits in using ChatGPT to promote the use of self-regulated learning skills. For instance, all students identified that using ChatGPT helped them connect new and prior knowledge (i.e., elaboration), identify concepts they didn't understand well (i.e., metacognition), and treat the initial ChatGPT-generated code as their starting point, rather than simply accepting it as it was (i.e., critical thinking). This aligns with previous research indicating that particularly participants with lower programming experience could benefit from regulatory support (Silva et al., 2024). However, it is important to note that not all aspects of critical thinking were equally promoted. While students felt confident using ChatGPT-generated code as a starting point, they reported that using ChatGPT did not promote them to question how convincing the ChatGPT-generated code was, seek supporting evidence, or consider alternative solutions. In the future, this could be better supported by embedding self-regulated learning scaffolds in the assessment task and explicit self-regulated learning instruction (Broadbent et al., 2020).

Participants reported two main challenges associated with using ChatGPT for code review education. First, using ChatGPT can hinder skill development. One student with some programming experience expressed concern that reliance on ChatGPT's automated code generation limited their ability to improve their own coding abilities. This student emphasised the need to eventually work without ChatGPT to avoid dependency and ensure they could write code independently at a comparable level (Case 1). While ChatGPT is useful for reducing cognitive load (in this case, intentionally so students could focus on code review rather than programming), it may be perceived by students as a missed learning opportunity. This suggests that, like educators and researchers (Lodge et al., 2023), students are also concerned about how and when generative artificial intelligence is integrated into their teaching and learning activities, ensuring it does not detract from their learning experience. Future implementations should explore alternative solutions for this issue, such as making explicit to students the intent of using ChatGPT for specific parts of the assessment.

The second challenge was related to foundational knowledge gaps, with the participant with minimal programming experience (Case 3) reporting that their lack of foundational knowledge made it difficult to fully benefit from using ChatGPT for code review education. Despite ChatGPT providing answers, the student often required additional clarification and explanation to understand the proposed solutions. This underscores the importance of having foundational knowledge in programming to effectively use generative AI tools like ChatGPT for certain educational purposes. Additionally, although two of the participants had some experience programming, they could still be considered somewhat novices in the field (i.e., self-reported programming experience < 5 on a 10-point scale). This could mean that their lack of knowledge or experience may have impacted their ability to compare the code to certain standards, not allowing them to evaluate the code appropriately (i.e., evaluative judgement; Tai et al., 2018). If they are novices, they may require additional support or resources to help them better understand standards and quality in their field. Further research is needed to better understand the threshold of initial knowledge required for effective interactions with generative AI in domain-specific educational contexts.

Although the findings from the current study are insightful and supportive of previous research, they should be interpreted with caution. Due to the limited sample size, the study did not achieve saturation of the qualitative data, suggesting that the results may not capture the full range of experiences and perspectives. Further research with a larger and more diverse sample is required to validate these findings and to explore additional factors that may influence the effectiveness of ChatGPT in code review education. Despite these limitations, the study provides valuable preliminary insights into the potential benefits and challenges of integrating ChatGPT into educational settings, highlighting important areas for future investigation.

Conclusion

The current study explored the use of ChatGPT to promote code review and self-regulated learning, revealing significant insights and identifying both benefits and challenges. Participants with varying levels of programming experience provided diverse perspectives on the quality and usefulness of ChatGPT-generated code. Novice programmers benefited from a deeper understanding of code interactions, while more experienced programmers found ways to improve the original code through continuous prompt optimisation. However, identified challenges included students' concerns about generative AI not detracting from their learning experience, and foundational knowledge gaps hindering their ability to interact with ChatGPT.

Overall, ChatGPT shows promise as a supportive tool for enhancing code review education, particularly for novice programmers. Its ability to promote self-regulated learning strategies, such as elaboration and metacognition, underscores its potential educational value. Future research should aim to validate these findings with larger samples and address the highlighted challenges to optimise the integration of generative AI tools like ChatGPT in educational settings. This study contributes valuable preliminary insights into the role of generative AI in programming education, setting the stage for further investigation and development.

Reference list

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77–101. https://doi.org/10.1191/1478088706qp063oa

Broadbent, J., Panadero, E., Lodge, J. M., & de Barba, P. (2020). Technologies to enhance self-regulated learning in online and computer-mediated learning environments. *Handbook of research in educational communications and technology,* 37-52. https://doi.org/10.1007/978-3-030-36119-8_3

Chong, C. Y., Thongtanunam, P., & Tantithamthavorn, C. (2021). Assessing the students' understanding and their mistakes in code review checklists: an experience report of 1,791 code review checklist questions from 394 students. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training* (ICSE-SEET) (pp. 20-29). https://doi.org/10.1109/ICSE-SEET52601.2021.00011

Daschner, S. (2022, December 15). *Coding Session With ChatGPT - How Helpful Is It?* [Video]. YouTube. https://www.youtube.com/watch?v=90JQkqZ59QU&t=350s

Dunsmore, A., Roper, M., & Wood, M. (2003). The development and evaluation of three diverse techniques for object-oriented code inspection. *IEEE transactions on software engineering, 29*(8), 677-686. https://doi.org/10.1109/TSE.2003.1223643

Feigenspan, J., Kästner, C., Liebig, J., Apel, S., & Hanenberg, S. (2012). Measuring programming experience. In *IEEE international conference on program comprehension (ICPC)* (pp. 73-82). IEEE. https://www.cs.cmu.edu/~ckaestne/pdf/icpc12.pdf

Hyytinen, H., Holma, K., Toom, A., Shavelson, R. J., & Lindblom-Ylänne, S. (2014). The Complex Relationship between Students' Critical Thinking and Epistemological Beliefs in the Context of Problem Solving. *Frontline Learning Research, 2*(5), 1-25. https://eric.ed.gov/?id=EJ1090929

Indriasari, T. D., Luxton-Reilly, A., & Denny, P. (2020). A review of peer code review in higher education. ACM Transactions on Computing Education (TOCE), 20(3), 1-25. https://doi.org/10.1145/3403935

Lee, I., & Hahn, S. (2024). On the relationship between mind perception and social support of chatbots. *Frontiers in Psychology, 15*, 1282036. https://doi.org/10.3389/fpsyg.2024.1282036

Lee, S., Lee, N., and Sah, Y. J. (2020). Perceiving a mind in a chatbot: effect of mind perception and social cues on co-presence, closeness, and intention to use. *Int. J. Hum. Comput. Interact. 36,* 930–940. https://doi.org/10.1080/10447318.2019.1699748

Lin, W., Garcia, M., Nasir, M., & Sultana, A. (2024, March). Beyond Traditional Teams: Using ChatGPT to Simulate Project Management Dynamics and Software Development in Online Higher Education. In J. Cohen & G. Solano (Eds.), *Proceedings of Society for Information Technology & Teacher Education International Conference* (pp. 2123-2129). Las Vegas, Nevada, United States: Association for the Advancement of Computing in Education (AACE). Retrieved June 12, 2024 from https://www.learntechlib.org/primary/p/224269/

Liu, Y., Le-Cong, T., Widyasari, R., Tantithamthavorn, C., Li, L., Le, X. B. D., & Lo, D. (2023). Refining ChatGPT-generated code: Characterizing and mitigating code quality issues. *ACM Transactions on Software Engineering and Methodology.* https://doi.org/10.1145/3643674

Lodge, J. M., de Barba, P., & Broadbent, J. (2023). Learning with generative artificial intelligence within a network of co-regulation. *Journal of University Teaching and Learning Practice, 20*(7), 1-10. https://doi.org/10.53761/1.20.7.02

Meijs, C., Neroni, J., Gijselaers, H. J., Leontjevas, R., Kirschner, P. A., & de Groot, R. H. (2019). Motivated strategies for learning questionnaire part B revisited: New subscales for an adult distance education setting. *The internet and higher education, 40*, 1-11. https://doi.org/10.1016/j.iheduc.2018.09.003

Mollick, E. R., & Mollick, L. (2023). Assigning AI: Seven approaches for students, with prompts. *SSRN.* https://ssrn.com/abstract=4475995

OpenAI. (2024). ChatGPT (Jun 13 version) [Large language model]. https://chat.openai.com/chat

Panadero, E. (2017). A review of self-regulated learning: Six models and four directions for research. *Frontiers in Psychology, 8*. https://doi.org/10.3389/fpsyg.2017.00422

Panadero, E., Alqassab, M., Fernández Ruiz, J., & Ocampo, J. C. (2023). A systematic review on peer assessment: intrapersonal and interpersonal factors. *Assessment & Evaluation in Higher Education, 48*(8), 1053-1075. https://doi.org/10.1080/02602938.2023.2164884

Pintrich, P. R., Smith, D. A., García, T., & McKeachie, W. J. (1991). *A manual for the use of the Motivated Strategies for Learning Questionnaire (MSLQ).* Retrieved June 13, 2024 from https://eric.ed.gov/?id=ED338122

Silva, L., Mendes, A., Gomes, A., & Fortes, G. (2024). What Learning Strategies are Used by Programming Students? A Qualitative Study Grounded on the Self-regulation of Learning Theory. *ACM Transactions on Computing Education, 24*(1), 1-26. https://doi.org/10.1145/3635720

Tai, J., Ajjawi, R., Boud, D., Dawson, P., & Panadero, E. (2018). Developing evaluative judgement: enabling students to make decisions about the quality of work. *Higher education, 76*, 467-481. https://doi.org/10.1007/s10734-017-0220-3

Yam, K. C., Bigman, Y. E., Tang, P. M., Ilies, R., De Cremer, D., Soh, H., et al. (2021). Robots at work: people prefer—and forgive—service robots with perceived feelings. *J. Appl. Psychol. 106*, 1557–1572. https://doi.org/10.1037/apl0000834

Walker, K. L., Ness, S., Reed, F., & Strang, K. (2021). A safe space: practicing teaching skills with avatars. In *Implementing augmented reality into immersive virtual learning environments* (pp. 120-134). IGI Global. https://doi.org/10.4018/978-1-7998-4222-4.ch007

Waseem, M., Das, T., Ahmad, A., Liang, P., Fahmideh, M., & Mikkonen, T. (2024). ChatGPT as a Software Development Bot: A Project-Based Study. In H. Kaindl, M. Mannion, & L. Maciaszek (Eds.), ENASE 2024: *Proceedings of the 19th International Conference on Evaluation of Novel Approaches to Software Engineering* (pp. 406-413). SCITEPRESS - Science and Technology Publications. https://doi.org/10.5220/0012631600003687

de Barba, P., & Tantithamthavorn, C. (2024). Challenges and opportunities in using ChatGPT as a team member to promote code review education and self-regulated learning. In Cochrane, T., Narayan, V., Bone, E., Deneen, C., Saligari, M., Tregloan, K., Vanderburg, R. (Eds.), *Navigating the Terrain: Emerging frontiers in learning spaces, pedagogies, and technologies*. Proceedings ASCILITE 2024. Melbourne (pp. 108-117). https://doi.org/10.14742/apubs.2024.1152

Note: All published papers are refereed, having undergone a double-blind peer-review process.