

# Integrating Conversational Large Language Models into Student Learning: A Case Study of ChatGPT in Software Engineering Education

Yi Liu

Department of Computer Science  
University of North Carolina Wilmington  
Wilmington, North Carolina, USA  
liuyi@uncw.edu

**Abstract**—This innovative practice full paper describes a pilot study exploring the integration of ChatGPT, a Conversational Large Language Model (LLM), into the student learning process in software engineering education, which emphasizes principles and methodologies in software development. Focused on a software engineering class, the study examines ChatGPT as a tool for problem clarification, modeling assistance, system design feedback, and implementation support in a project on modeling, designing, and implementing a solution using finite state processes and concurrent programming in Java. A survey designed for the case study collects insights into students' experiences with ChatGPT at different stages of the project. Student feedback on using ChatGPT and their performance on the project are analyzed to understand the impact of conversational LLMs on learning outcomes and to address whether there is room for improvement in enhancing the use of conversational LLMs in software engineering education.

**Index Terms**—Conversational LLMs, ChatGPT, Software engineering education

## I. INTRODUCTION

Conversational Large Language Models (LLMs) have gained significant attention for their wide usage in natural language processing (NLP) tasks such as translation, sentiment analysis, and chatbots. One of the most popular LLM-based AI chatbots is ChatGPT, developed by OpenAI. Since its first release to the public in November 2022, ChatGPT has presented its versatile capabilities for tasks like answering questions, language translation, creative writing, and code generation [1].

In education, such generative AI tools have the potential to revolutionize learning, for example, in personalized learning, language instruction, and feedback applications [2]. As indicated in [3] [4], these tools can benefit higher education by streamlining administrative tasks, enhancing teaching through personalized feedback, aiding research, and improving student engagement. However, concerns have been raised regarding privacy, biases, overreliance on AI, and academic dishonesty. These issues call the need for critical thinking and human oversight in the use of such tools.

Software engineering education emphasizes teaching students the principles, methodologies and technique in software development process, such as requirements analysis, design,

implementation, verification and validation, and maintenance. The overarching *learning outcomes for students* include the ability to apply these principles to solve real world problems. The goal of this pilot study is to investigate the feasibility of integrating conversational LLMs, specifically ChatGPT, into the student learning process in software engineering education. Focused within the context of a software engineering class, the study explores ChatGPT as a tool for problem clarification, modeling assistance, system design feedback, and implementation support in a project on modeling, designing and implementing a solution using finite state processes and concurrent programming in Java. The study includes the design of a survey to collect students' feedback on their experiences using ChatGPT at different stages of the project. The study analyzes the student feedback and their performance on the project to understand the impact of conversational LLMs on the learning outcomes. The study answers the following two research questions:

- R1: Does the integration of ChatGPT, have a noticeable impact on student learning outcomes in a software engineering class, specifically in problem clarification, modeling assistance, system design feedback, and implementation support?
- R2: Is there room for improvements in the integration of conversational LLMs in student learning process, based on insights from student feedback and their performance?

The rest of the paper is organized as follows. Section II provides a brief overview of ChatGPT and discussed related work of applying ChatGPT in coursework. Section III presents the design and implementation of applying ChatGPT in a software engineering class to support the student learning process in problem clarification, modeling assistance, system design feedback, and implementation in a project. Section IV analyzes the performance outcomes and students' feedback, and answers the two research questions. Section VII offers the summary of the work and outlines the future research direction.

## II. BACKGROUND

### A. ChatGPT

ChatGPT is a large language model developed by OpenAI based on the GPT (Generative Pre-trained Transformer) architecture. It uses self-attention mechanisms and unsupervised pre-training on vast amounts of text data from the internet to generate human-like text based on the input it receives [1] [5]. Users interact with ChatGPT by providing prompts, which are initial inputs that guide the model's responses. With the ability to understand and generate natural language, ChatGPT has shown remarkable performance in a wide range of tasks, such as answering questions, writing essays, and generating code. ChatGPT 3.5 is the latest model open to the public without charge, while ChatGPT 4 offers enhanced capabilities with a fee.

### B. Related work

There are some educational studies that have applied ChatGPT in engineering courses.

In [6], the authors explore the use of ChatGPT as a teaching aid in signal processing classes and find that while ChatGPT can provide explanations, generate code snippets, and assist with problem-solving, it may struggle with advanced mathematical concepts and produce inconsistent or incorrect outputs. The authors then suggest using ChatGPT for tasks such as generating practice problems, explaining concepts in different ways, and providing initial code templates.

In [7], the authors present their study of using ChatGPT and DALL-E to assess and enhance student understanding in structural engineering classes. Students describe engineering concepts or problems, which the AI models analyze to provide feedback on conceptual understanding and identify misconceptions. DALL-E generates visual representations of these descriptions. The authors find that such approach promotes active learning, personalized feedback, and clearer communication of technical concepts. However, they also note that AI models can generate inaccurate outputs, needing human oversight.

In [8], the authors apply ChatGPT into two computer science courses, database systems and mobile/smart devices & technology, combined with evidence-based learning strategies including well-defined learning goals and strategic assessment. In the database systems class, students use ChatGPT to create a database for a bookstore, and in the mobile technology class, they use ChatGPT to create a Java-based user interface to interact with a web-based API. The authors argue that while generative AI can be a powerful aid, it brings risks such as students becoming overly reliant on it, potential biases and inaccuracies, and the need for human oversight.

Different from these works, this study explores the use of ChatGPT to assist students in achieving learning outcomes in software development processes, including requirements clarification, design, and implementation.

## III. THE DESIGN OF THE STUDY

In this study, the experiment was conducted in the CIS 461 Formal Methods for Software Engineering class, taught by the author in the Department of Computer & Information Science at the University of Massachusetts Dartmouth in Fall 2023.

The course was offered as a mandatory class for software engineering option students and as a technical elective for both undergraduate and graduate computer science students. In Fall 2023, the course had a total of 30 students, including 15 undergraduates and 15 graduates.

Formal Methods for Software Engineering introduces students to the application of formal model-based methods in software development, which includes applying formal languages and techniques to analyze and verify specifications and designs, and implementing these designs into applications. Upon successful completion of this class, students are expected to achieve the following learning outcomes: (1) apply formal model-based methods in software development, (2) analyze problems and model solutions with Finite State Machines [9], and (3) implement models using concurrent Java programming.

In this study, students were assigned a small project to use ChatGPT as a tool for problem clarification, modeling assistance, system design feedback, and implementation support. The project was offered with bonus credits, allowing students to choose whether to complete it or not. As a result, a total of 27 students participated, including 13 undergraduates and 14 graduates.

### The problem:

The project presents a *producer and consumer* scenario, a classical synchronization problem in operating systems and concurrent programming [10]. The problem statement is given below.

```
There are 3 bees and a bear who share a pot of honey.
The pot is initially empty, and its capacity is H
portions of honey (set H=3 in the model). The bear
waits until the pot is full, eats all the honey, and
then waits until the pot is full again. Each bee
repeatedly gathers one portion of honey and puts it
into the pot. if the pot is full when a bee arrives,
it must wait until the bear has eaten the honey
before adding its portion. This repeats forever.
```

### The requirements:

The students were required to (1) model the solution to the Bear and Bee problem using Finite State Processes (FSP) [11], and (2) implement the developed FSP model in Java and thoroughly test it.

The students were *encouraged to use ChatGPT 3.5 to enhance their understanding of the problem, refine their model, and get assistance with debugging*. The following guidance was given as a clarification:

- Problem clarification: ask ChatGPT to clarify the bear and bee problem by providing several examples.
- Modeling assistance: describe current progress in modeling to ChatGPT. For example, ask ChatGPT to suggest

potential processes if not sure where to start; or ask ChatGPT for suggestions on refining a developed a model.

- System design: share system design with ChatGPT and ask for feedback.
- Implementation assistant: for example, share a snippet of your Java code and ask ChatGPT for feedback; or ask ChatGPT for assistance in debugging.

Two rules were specifically given to the students for their communication with ChatGPT:

- Students should not seek direct implementation from ChatGPT.
- Students should use critical thinking skills to evaluate the feedback from ChatGPT.

#### Deliverables:

The students were required to submit the designed FSP model for the problem, the Java source code that implemented the FSP model, the chat record with ChatGPT, and a completed survey of 10 questions.

## IV. RESULTS

The outcomes of using ChatGPT to support students' learning are based on two factors: the students' performance on the project and their feedback on the experience of using ChatGPT.

### A. Sample chats

Fig. 1 illustrates a chat on problem clarification that one student submitted. In this chat, the student provided the problem description and requested ChatGPT to provide several traces of the process. ChatGPT provided one possible trace showing how the bees produce honey into the pot and how the bear consumes it from the pot. ChatGPT provided a correct trace for this problem.

Fig. 2 shows a chat on FSP modeling assistance. The student provided the designed FSP model and asked ChatGPT to provide feedback. ChatGPT initially offered very general suggestions, which were omitted from the figure due to the length limit of the paper, and then provided a revision to the model. However, the model ChatGPT provided was identical to the original model, with no changes made.

In Fig. 3, the student illustrated the idea of how to implement the model as monitors or processes and requested ChatGPT to provide suggestions. Following the rule of not eliciting direct implementation from ChatGPT, the student specifically asked ChatGPT not to provide Java code. ChatGPT agreed with the student's idea on assigning the monitor and processes and also provided some explanation.

### B. Students' performance

The criteria for assessing the students' performance in this project were aligned with our department's ABET assessment criteria. Satisfactory performance was defined as 80% of students achieving a score of 80/100 or higher in both modeling and implementing the project problem.

Fig. 4 reports the distribution for the 27 students' performance on modeling solutions with FSP. 19 students (67%)

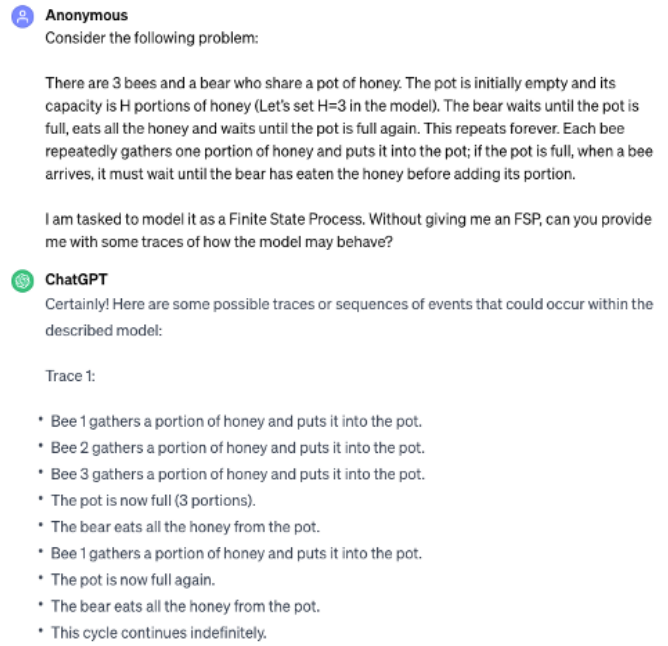


Fig. 1: Chat on problem clarification



Fig. 2: Chat on FSP modeling assistance

**Anonymous**  
I now have to implement this model using Java code. Please do not give me the full Java code in your answer. I was thinking about modeling Bee and Bear as processes and Pot as a monitor. Is this a good idea?

**ChatGPT**  
Yes, using a monitor to represent the Pot and modeling the Bee and Bear as separate processes in Java is a reasonable approach. Monitors can help control access to shared resources (in this case, the Pot) and ensure synchronization between different threads or processes.

Here's a basic outline of how you might structure your Java implementation:

1. **Bee Class:** Define a `'Bee'` class that represents the behavior of a bee. Each `'Bee'` object could be a thread or a process that gathers honey and puts it into the Pot.
2. **Bear Class:** Create a `'Bear'` class that models the behavior of the bear. Similar to the `'Bee'` class, each `'Bear'` object could be a thread or a process that consumes honey from the Pot.
3. **Pot Monitor:** Implement a `'PotMonitor'` class that encapsulates the behavior of the Pot. This monitor should provide methods for the bees to add honey (`'putHoney'`) and for the bear to consume honey (`'eatHoney'`). Ensure proper synchronization mechanisms (like `'synchronized'` blocks, locks, or semaphores) to control access to the Pot and manage concurrent interactions between bees and the bear.
4. **Main Program:** Create a main program or controller that initializes the Pot, bees, and bear objects, starts their respective threads or processes, and coordinates their interactions based on the defined behavior.

Fig. 3: Chat on system design assistance

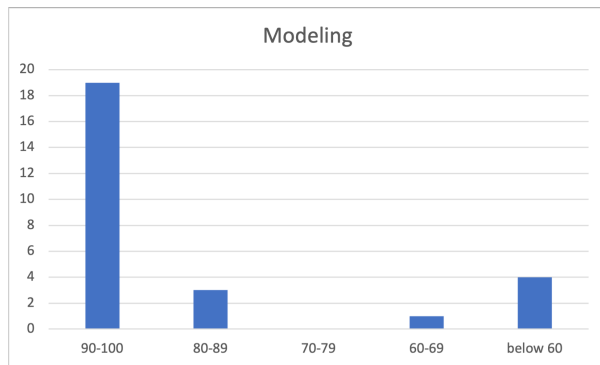


Fig. 4: Grades distribution on modeling solutions

received a score of 90/100 or higher, and 3 (19%) scored between 80 and 90. Therefore, 81% of students achieved a score of 80/100 or higher.

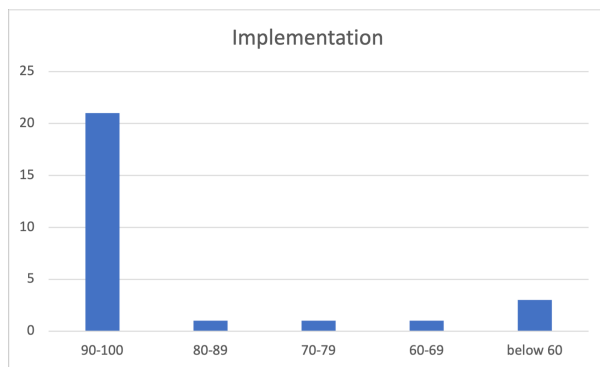


Fig. 5: Grades distribution on implementation

Fig. 5 illustrates the students' performance in implementing the FSP model using Java concurrent constructs. 21 students

(67%) scored 90/100 or higher, and 1 student (19%) scored between 80 and 90. Similarly, 81% of students achieved a score of 80/100 or higher.

### C. Students' feedback

A survey aimed at collecting students' opinions and experiences with the use of ChatGPT in this project was given to the students, and the feedback was collected along with the rest of their submissions. The survey consisted of 7 Likert scale questions (Q1, Q2, Q4, Q6, Q8, Q9, and Q10) and 4 open-ended question (Q3, Q5, Q7, and Q11), as described below.

- 1) How well did ChatGPT understand your questions related to the bear and bee problem?
- 2) How helpful was ChatGPT in providing examples to clarify the bear and bee problem?
- 3) Were the examples provided by ChatGPT helpful in enhancing your understanding of the problem?
- 4) How helpful was ChatGPT in providing feedback to your model design?
- 5) Did you adopt ChatGPT's suggestions in refining your model? Why?
- 6) How useful was ChatGPT in providing feedback on your system design idea?
- 7) Did the feedback from ChatGPT help improve your system design?
- 8) How helpful was ChatGPT in providing feedback on your Java code snippet?
- 9) How helpful was ChatGPT in assisting with debugging?
- 10) Overall, how satisfied are you with using ChatGPT as a tool for assistance in the bear and bee problem?
- 11) Please share any additional comments, suggestions, or insights regarding your experience using ChatGPT for this assignment.

Fig. 6 illustrates the survey results for the likert scale questions in the form of charts.

**Q1, Q2, and Q3:** These three questions collected students' feedback on the *Problem clarification* task using ChatGPT, assessing whether ChatGPT helped them understand the problem better.

In Q1, 63% of students agreed that ChatGPT was "excellent" or "good" at understanding their questions related to the given problem, while 33% thought it was "fair". In Q2, 56% of students found that ChatGPT provided "very helpful" or "helpful" examples to clarify the bear and bee problem, while 41% of students thought it was fair. 4% of the students rated ChatGPT poorly in either understanding their questions or providing examples.

22 students responded to the open-ended question Q3 regarding whether the examples provided by ChatGPT were helpful in enhancing their understanding of the problem. Of these, 16 gave positive feedback. For example, one student said, "Yes. I was able to understand the problem better and the examples were simple and easy to grasp"; Three students indicated that they thought the examples were somewhat helpful, while two believed they were not very helpful since they already understood the problem well before asking ChatGPT.

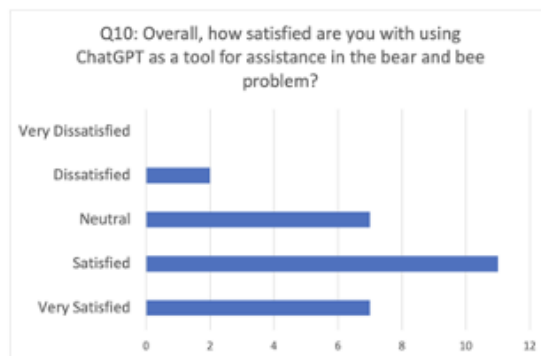
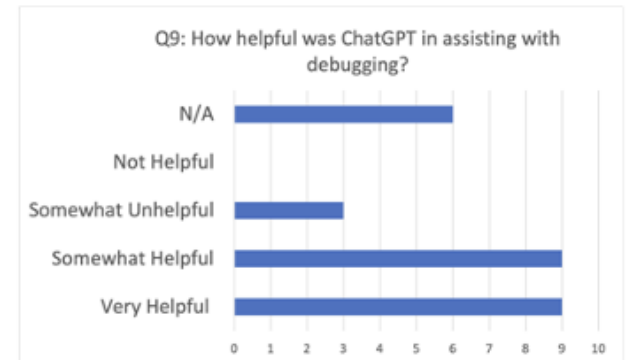
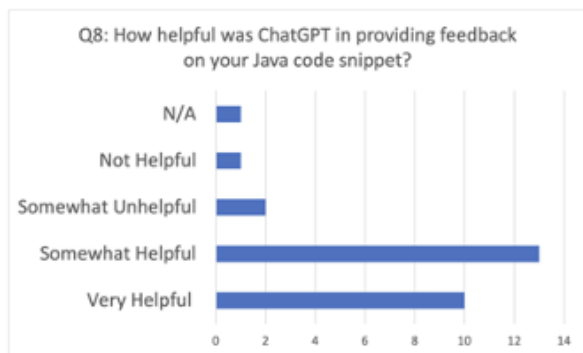
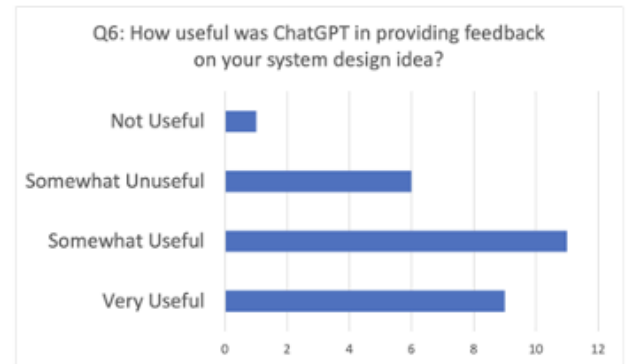
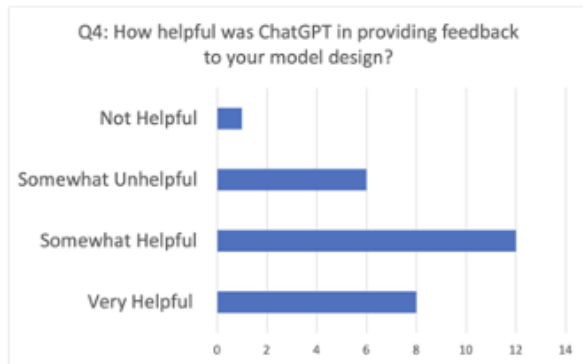
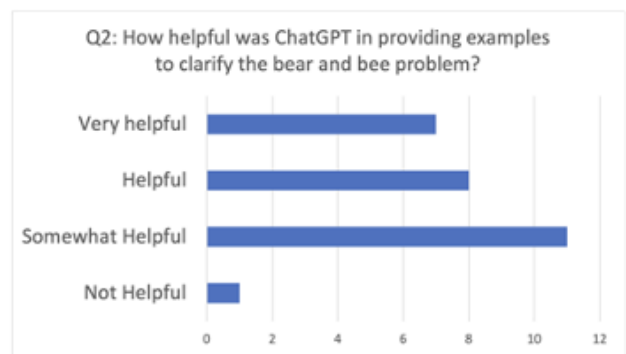
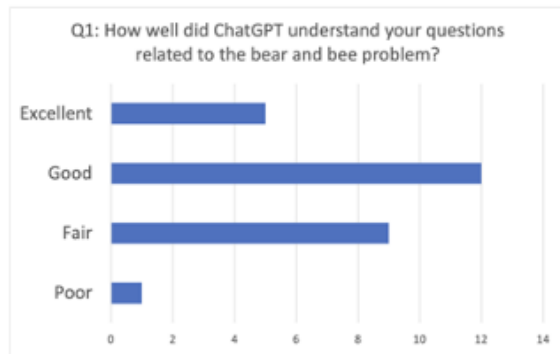


Fig. 6: Survey results for likert scale questions

For instance, one student stated, *"If I did not understand the problem at all going in, it would have been somewhat helpful to me,"* and another mentioned, *"They're basic, and it didn't tell me anything that I already know."* One student claimed that *"there were no examples provided."*

**Q4 and Q5:** These two questions asked the students whether they thought ChatGPT provided helpful *Modeling assistance* for their FSP model designs.

Regarding Q4, 74% of students believed that ChatGPT provided very helpful or helpful feedback to their model designs, which 22% thought it was somewhat unhelpful, and 4% considered ChatGPT's feedback not helpful.

When asked whether they adopted ChatGPT's suggestions to refine their FSP models (Q5), out of the 24 students who responded, 15 students said "no," and 9 students said "yes." Some reasons why students did not adopt ChatGPT's suggestions include, "I did not think that ChatGPT's model was correct, so I used my critical thinking"; "It didn't really make any good suggestions; it basically just repeated the model I gave it"; and "No, because ChatGPT is not great at FSP modeling." Conversely, reasons why students adopted the suggestions include, "Yes, it told me to adjust my indexes for better clarity"; "I used its suggestion to add more comments"; and "Yes, the suggestions helped me fix the model."

**Q6 and Q7:** These two questions addressed the students' opinions on whether ChatGPT provided helpful feedback on their *System design* based on their FSP models.

In the Likert scale question Q6, 74% of students agreed that ChatGPT provided very helpful or helpful feedback to their system designs, which 22% thought it provided somewhat not useful feedback, and 4% considered ChatGPT's feedback was not helpful.

Regarding the open-ended question about whether ChatGPT's feedback helped improve their system designs, 25 students responded. Among them, 14 provided positive feedback. For example, one student said, *"Yes, the feedback from ChatGPT did improve the design of my system; it allowed my program to run efficiently."* Another student mentioned, *"Yes, it helped on which processes are supposed to be implemented as threads and which as a monitor."*

However, 10 students provided negative feedback. For instance, one student stated, *"No. It just confirmed my original idea."* Another student commented, *"My system design was already solid as per ChatGPT's feedback."*

**Q8 and Q9:** To address the *Implementation assistant*, these two questions asked the students' opinions on whether ChatGPT provided useful feedback on their Java implementations and helped them with debugging.

For Q8, 85% of students found ChatGPT to be very helpful or helpful in providing feedback on their Java code snippets, while 7% found it somewhat helpful and 4% found it not helpful. In addition, 4% of students chose N/A since they did not use ChatGPT for checking their implementations.

For Q9, 67% of students believed ChatGPT was very helpful or helpful in assisting with debugging, while 11% thought it was somewhat helpful. Similarly, 22% of students chose N/A

since they did not use ChatGPT for helping them debug their code because they did not have difficulties in coding.

**Q10 and Q11:** These two questions gathered the students' feedback on their overall satisfaction with ChatGPT as a tool to assist them in completing the project.

In Likert scale question Q10, 66% of students indicated that they were either very satisfied or satisfied with the assistance from ChatGPT with the Bee and Bear problem, 26% stated they were neutral, and 7% considered themselves dissatisfied with the assistance.

17 students responded to the open-ended question Q11, providing additional comments on their experience with ChatGPT in this project. The responses fell into three categories:

- Very positive feedback, such as one student stating, *"I love this tool, as someone who has such a hard time coding anything at all, it helped steer me in the right direction,"* and another saying, *"ChatGPT is a great tool for programmers to get another opinion or even check their work. In my experience, ChatGPT has been very helpful in answering simple questions and providing helpful feedback on code and debugging."*
- Positive feedback with concerns, for example, one student expressed, *"I feel ChatGPT helped me break down the problem and understand it. However, the FSP solution which was provided by it was not fully functional. ChatGPT was useful with the Java code part."*
- Negative feedback about their experience, for instance, a student stated, *"I was always worried if ChatGPT would accidentally give me the direct implementation of either my FSP or Java code, so I had to keep restating to not do that. It's better to ask for feedback from a teacher since I know that wouldn't happen."*

## V. DISCUSSION

The students' performance and their feedback described in subsections IV-B and IV-C serve as the basis for answering the research questions raised in Section I.

A. R1: Does the integration of ChatGPT, have a noticeable impact on student learning outcomes in a software engineering class, specifically in problem clarification, modeling assistance, system design feedback, and implementation support?

The students' performance described in IV-B indicated that 81% of them achieved 80/100 or higher in both modeling the solutions for the given project and implementation of the models in Java. Thus, the students' performance met the satisfaction criteria.

The students' feedback on the survey questions, showed in IV-C, shows that the majority of the students agreed that ChatGPT provided helpful assistance in question clarifications, modeling, reviewing the system design, and providing feedback on implementations. Overall, their experience with using ChatGPT for the project was satisfactory.

These results provide a positive answer to R1, indicating that in this pilot study, we observed a positive impact of using ChatGPT in supporting learning objectives

*B. R2: Is there room for improvements in the integration of conversational LLMs in student learning process, based on insights from student feedback and their performance?*

An analysis was conducted on students' answers to the project questions (students' performance), the chats students submitted, and their feedback. The observation is described below.

1) *Prompt design makes a difference:* In the context of conversations between people and ChatGPT, a *prompt* is the input given to ChatGPT to elicit a specific response or generate relevant information. It is important to make a prompt clear and understandable since the quality and relevance of ChatGPT's response depend significantly on how well the prompt conveys the user's intent and context.

The project description did not provide any exemplary prompts for the students to use when interacting with ChatGPT. Designing an effective prompt was expected as part of the students' experience in this project.

When reviewing the chats for clarifying the project requirements with examples and the students' feedback to survey questions Q1, Q2, and Q3, we found that the same individual who chose "Poor" in Q1 and "Not Helpful" in Q2 and Q3 fed the entire project description, including the problem statement, requirements, and deliverables, to ChatGPT. Consequently, ChatGPT became confused and provided a summary of the project description instead of giving examples.

2) *Refinements to the prompts are needed for better conversation results:* Sometimes ChatGPT does not provide sufficient or fully relevant answers to a prompt, in which case refining the prompt with a follow-up question is necessary, much like in human conversations.

For example, in the case above, where the student copied the entire project description in one prompt and ChatGPT got confused, if the student had followed up with a prompt saying, 'Please provide three examples according to the problem statement given above,' the student would have had a better chance of receiving relevant examples from ChatGPT.

3) *ChatGPT is pre-trained on a vast corpus of general text, which may not be optimized for specialized tasks or domain-specific data:* The majority of the students (15 out of 24 in answering Q5) stated that they did not adopt ChatGPT's suggestions for refining their FSP models. Their comments were consistent: ChatGPT was not good at FSP modeling.

Built upon a pre-trained large language model trained on a vast corpus of general text, ChatGPT is highly capable of responding to questions where a more extensive corpus has been involved in the training. For example, students have found ChatGPT to be helpful in debugging Java code. Java, being a popular programming language, has a good amount of online tutorials, example code, and projects implemented in Java available in public repositories like GitHub, which serve as sources to train the GPT model for Java programming. In contrast, FSP, a modeling language used solely for formal specifications, has limited online or public sources available, providing less material to train GPT models to understand it.

Based on the analysis, the answer to R2 is positive; yes, we believe there is room for improvement in the integration of conversational LLMs in the student learning process in a study like this.

Firstly, to address observations (1) and (2), clear instructions and guidelines on interacting with ChatGPT, including example prompts, should be provided to the students. This will help them understand the importance of forming prompts and how to come up with effective ones.

Secondly, to address observation (3), two options can be adopted to improve the customization of ChatGPT to fit our needs, in this study, better support on FSP modeling. One approach would be to fine-tune GPT 3.5 for this specific purpose [12] [13], and the other would be to customize a specific GPT tool from GPT 4 [14].

## VI. THE LIMITATION OF THE STUDY

This study aims to investigate the impact of using a conversational Large Language Model (LLM) in assisting students with a software engineering project. The study design involves all students in the course using ChatGPT as an assistant in the project so that each participant has an opportunity to experience and engage with ChatGPT throughout the stages of problem-solving, modeling, system design, and implementation.

However, a limitation of this study is the absence of a control or comparison group. In a typical experimental design, a control group would perform the project without the assistance of ChatGPT, which allows a direct comparison between the performance and learning outcomes of those who used ChatGPT and those who did not. Without such a comparison group, it is difficult to determine whether the performance with the assistance of ChatGPT is better than without it, and whether similar results could have been achieved through other forms of support, such as peer collaboration, instructor guidance, or independent problem-solving.

## VII. CONCLUSION

This pilot study explores ChatGPT as a tool for problem clarification, modeling assistance, system design feedback, and implementation support in a software engineering project. Both students' performance and feedback indicated positive results from using ChatGPT to support learning objectives in this study.

This study serves as groundwork for integrating conversational LLMs into the student learning process, and its findings aim to enhance the design and expansion of incorporating conversational LLMs like ChatGPT in software engineering education.

Future work will focus on two directions: firstly, extending the study to more software engineering classes, such as those on software design and software development; secondly, developing GPT tools tailored to specific software engineering topics in each class by fine-tuning GPT 3.5.

## REFERENCES

- [1] K. Hines, "History of chatgpt: A timeline of the meteoric rise of generative ai chatbots," Jun. 2023. [Online]. Available: <https://www.searchenginejournal.com/history-of-chatgpt-timeline/488370/>
- [2] T. Adıgüzel, M. H. Kaya, and F. K. Cansu, "Revolutionizing education with ai: Exploring the transformative potential of chatgpt," *Contemporary Educational Technology*, vol. 15, no. 3, p. ep429, 2023.
- [3] J. Dempere, K. Modugu, A. Hesham, and L. K. Ramasamy, "The impact of chatgpt on higher education," *Frontiers in Education*, vol. 8, p. 1206936, 2023.
- [4] P. Rospigliosi, "Artificial intelligence in teaching and learning: what questions should we ask of chatgpt?" *Interactive Learning Environments*, vol. 31, no. 1, pp. 1–3, 2023.
- [5] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.
- [6] M. K. Banavar, L. Shri, N. Sparks, and A. Cohen, "Being brave in a new world: Leveraging chatgpt in signal processing classes," in *2023 IEEE Frontiers in Education Conference (FIE)*. IEEE, 2023, pp. 1–5.
- [7] R. A. Chacón Flores, C. Vieira, and H. Murzi, "Eliciting student understanding in structural engineering classrooms using natural language models and image generation," in *FIE 2023: College Station, TX, USA, October 18-21, 2023: conference proceedings*. Institute of Electrical and Electronics Engineers (IEEE), 2023, pp. 1–5.
- [8] P. Lauren and P. Watta, "Work-in-progress: Integrating generative ai with evidence-based learning strategies in computer science and engineering education," in *2023 IEEE Frontiers in Education Conference (FIE)*. IEEE, 2023, pp. 1–5.
- [9] F. Wagner, R. Schmuki, T. Wagner, and P. Wolstenholme, *Modeling Software with Finite State Machines: A Practical Approach*. CRC Press, 2006.
- [10] A. Downey, *The little book of semaphores*. Green Tea Press, 2008, vol. 2, no. 2.
- [11] J. Magee and J. Kramer, *Concurrency: State Models and Java Programs*. John Wiley & Sons, 2006.
- [12] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language models to follow instructions with human feedback," *Advances in neural information processing systems*, vol. 35, pp. 27 730–27 744, 2022.
- [13] S. Zhang, L. Dong, X. Li, S. Zhang, X. Sun, S. Wang, J. Li, R. Hu, T. Zhang, F. Wu, and G. Wang, "Instruction tuning for large language models: A survey," 2024.
- [14] OpenAI, "Gpt-4: Openai's latest language model," <https://www.openai.com/research/gpt-4>, 2023.