

# Tailoring Your Code Companion: Leveraging LLMs and RAG to Develop a Chatbot to Support Students in a Programming Course

Carlos Alario-Hoyos, Rebiha Kemcha, Carlos Delgado Kloos, Patricia Callejo<sup>1</sup>  
 Iria Estévez-Ayres, David Santín-Cristóbal, Francisco Cruz-Argudo, José Luis López-Sánchez  
*Universidad Carlos III de Madrid* Av. Universidad 30, E-28911 Leganés (Madrid), Spain  
 {calario@it, rkemcha@pa, cdk@it, pcallejo@it, ayres@it, dsantin@di, paco@di, jllsanch@di}.uc3m.es

**Abstract**—Students frequently rely on chatbots powered by generative Artificial Intelligence (GenAI), such as ChatGPT, Copilot, Gemini, and Claude, to assist with a wide range of academic tasks. However, these chatbots are not specifically designed for the context of particular courses, which can lead to responses that are sometimes inaccurate or insufficiently relevant. This paper introduces a chatbot specifically designed to support first-year engineering students in a Java programming course. Developed using the Retrieval-Augmented Generation (RAG) technique, the chatbot draws on course-specific resources such as videos, quizzes, programming exercises, and other materials, while using OpenAI’s Large Language Models (LLMs) GPT-4 and GPT-3.5 for information analysis and response generation. The data collected, consisting of logs from 1,059 messages sent by students to the chatbot and 30 responses to a survey, indicate that students primarily used the chatbot to clarify concepts and explain code snippets. Moreover, most of the students reported that the responses provided by the chatbot were well suited to the Java programming course.

**Index Terms**—Large Language Models (LLMs), Retrieval-Augmented Generation (RAG), Generative Artificial Intelligence (GenAI), Chatbots, and Programming Course.

## I. INTRODUCTION

Numerous efforts have been made to develop educational technologies that provide personalized support and feedback to students, complementing teacher guidance [1]. Intelligent Tutoring Systems (ITS) are designed for this role, using advances in artificial intelligence (AI), including natural language processing (NLP), machine learning (ML), and deep learning (DL) [2], [3]. In many cases, students interact with ITS through a text-based interface, allowing them to engage in conversations and receive customized assistance [4].

The development of conversational agents (i.e., chatbots) aimed at supporting and tutoring students in their learning process has primarily focused on tasks such as explaining and reviewing concepts, proposing exercises tailored to the student’s knowledge level [5], providing feedback on specific tasks, and guiding students through the steps needed to complete exercises [6], [7]. This approach requires extracting detailed information from the educational context of the student (e.g., course and topic) to inform the chatbot during its development process [8], allowing for more contextually relevant conversations.

The development of chatbots for educational purposes has been accelerated by the rapid advancements in Generative Artificial Intelligence (GenAI) [9], Large Language Models (LLMs) [10], and particularly following the release of ChatGPT by OpenAI on November 30, 2022 [11]. Since then, leading technology companies are having a strong competition to create advanced chatbots, such as Microsoft’s Copilot [12], Google’s Gemini [13], and Anthropic’s Claude, supported by Amazon [14]. Additionally, numerous applications have emerged, building on these chatbots or the LLMs they utilize, including GPT-3.5, GPT-4, Gemini 1.5 Pro, Claude 3 Opus, Claude 3 Sonnet, Llama 3, and all their variations.

The widespread availability of chatbots, many offering free access, has led a growing number of students to integrate these tools into their daily academic routines, using them for tasks such as completing assignments and resolving subject-related questions [15], [16]. However, effective use of these tools requires thoughtful interaction, including providing detailed context in the prompts to ensure clear communication [17]. However, many students may lack the necessary skills to create effective prompts when engaging with chatbots, which can limit the quality of the support they receive.

This paper presents the design and development of a chatbot specifically designed to assist first-year engineering students in a course on Java programming. This course is called Systems Programming and is offered in English and Spanish in various telecommunication engineering degrees at a Spanish university. The chatbot was built using OpenAI technologies (GPT-3.5 and GPT-4 LLMs), the Retrieval-Augmented Generation (RAG) technique [18], [19], and pre-existing course content, which had previously been created by the course instructors [20]. This content includes a series of MOOCs (Massive Open Online Courses) available on the edX platform in both languages, featuring videos, quizzes, programming exercises, and supplementary materials. These resources were used to generate embeddings for the RAG model, providing the LLMs with additional course-specific context during student interactions with the chatbot.

The remainder of this paper is structured as follows. Section II reviews related work on the use of chatbots to support students, with an emphasis on GenAI-powered chatbots. Sec-

tion III details the development of the chatbot, including its technical implementation and the web application provided for student interaction. Section IV outlines the methodology of the experiment conducted to assess the effectiveness of the chatbot in supporting students in the Java programming course; this section also describes the experimental scenario and the data collection instruments. Section V presents the findings of the study based on the data collected. Finally, Section VI draws conclusions and discusses potential directions for future work.

## II. RELATED WORK

Conversational agents, commonly known as chatbots, have been in existence for nearly six decades, ELIZA being one of the earliest examples [21]. These chatbots are designed to engage in human-like conversations using natural language. Early versions of these chatbots operated by recognizing keywords or phrases and generating pre-programmed responses. However, modern chatbots have advanced significantly by incorporating Natural Language Processing (NLP), Machine Learning (ML), and Deep Learning (DL) techniques [22]. A notable milestone was IBM Watson, an advanced system that answered complex queries in natural language, paving the way for Watson-powered chatbots in various applications. [23]. More recently, chatbots have begun using Generative AI (GenAI) models based on transformer-based deep neural networks [24], particularly Large Language Models (LLMs), with popular examples such as ChatGPT by OpenAI [11], Copilot by Microsoft [12], Gemini by Google [13], and Claude by Anthropic [14].

When designing a chatbot, several key criteria must be considered [25], [26], including *interaction mode*, *application*, *design approach*, and *knowledge domain*. The *interaction mode* refers to how the chatbot communicates with users, typically through text or voice. The *application* of a chatbot can be either task-oriented, focusing on specific tasks such as troubleshooting technical issues or assisting with hotel reservations, or non-task-oriented, aimed at general conversation or social interaction. The *design approach* can be rule-based, relying on predefined rules and pattern-matching techniques, or AI-powered, utilizing NLP, ML, and DL to generate more context-aware responses. Additionally, chatbots can be categorized by their *knowledge domain*: domain-specific chatbots are specialized in particular areas such as education or retail, while open-domain chatbots can engage in conversations on a wide range of topics.

Chatbots like ChatGPT, Copilot, Gemini, and Claude initially operated exclusively through text-based interaction, but have progressively evolved to support multimodal interaction. This allows them to recognize various input formats, such as images and voice, and deliver outputs in multiple formats, including both text, image, and voice. Additionally, their application is non-task-oriented, meaning they are designed for general conversation rather than specific tasks. Their design approach is AI-powered, specifically driven by Generative AI (GenAI), and they were developed as open-domain chatbots, capable of engaging with a broad range of topics.

The open-domain nature of chatbots like ChatGPT, Copilot, Gemini, and Claude can lead to inappropriate or inaccurate responses due to limited context. This is especially concerning in education, where some students may over-rely on them for academic support [27]. To address this, it is essential to develop chatbots that are tailored to specific knowledge domains (e.g., computer science, math) to offer more accurate and relevant assistance. Creating task-oriented chatbots could enhance their usefulness for tasks like explaining concepts, assisting with homework, or preparing for exams.

An approach to improving domain-specific chatbot performance is through fine-tuning techniques [28], where a pre-trained LLM is adapted to perform new, more specific tasks by following certain rules. However, fine-tuning can be computationally expensive. Alternatively, Retrieval-Augmented Generation (RAG) [18] can be used to incorporate relevant information, enabling chatbots to generate more accurate, context-aware responses. RAG allows dynamic access to external knowledge sources, reducing the need for extensive fine-tuning while still aligning responses with specific educational content.

The current literature offers several examples of GenAI and LLM-powered chatbots developed to support students in specific educational contexts or to perform targeted educational tasks. The CS50 “Duck” [29], part of Harvard University’s CS50.ai suite, is a chatbot designed to provide students with code explanations and course-related assistance. This suite includes the tool “Explain Highlighted Code,” which provides instant explanations for selected code snippets. The chatbot leverages OpenAI’s GPT-4 to simulate a one-on-one teacher-student interaction, guiding students through problems by offering support and prompts, rather than directly providing answers, thus encouraging independent problem-solving.

Other GenAI-powered chatbots include Khanmigo [30], Artemis [31] and AlgoBo [32]. Khanmigo, developed by Khan Academy, uses GPT-4 and applies the Socratic method to support students across various knowledge domains, encouraging critical thinking and problem-solving without directly revealing answers. Artemis is an Automated Programming Assessment System enhanced with an AI tutor powered by the GPT-3.5-Turbo model. It allows students to submit code and receive immediate, context-aware feedback, offering personalized, real-time assistance with programming assignments. AlgoBo, powered by GPT-4, is a chatbot designed to simulate common misconceptions and knowledge gaps. It works with TeachYou to help students solve algorithm-based programming problems, promoting a learning-by-teaching approach.

Finally, it is important to highlight the development of GPTs [33], which are custom versions of ChatGPT designed for specific purposes. This includes GPTs tailored to the educational field, focusing on specific tasks and knowledge domains (e.g., math). For instance, Tutor Me is a GPT developed by Khan Academy that serves as a personal tutor, modeled after Khanmigo but in a simpler form. Overall, the development of chatbots for specialized tasks and knowledge areas is essential, leveraging the potential of GenAI and LLMs to provide more targeted and effective educational support.

### III. DESIGN AND DEVELOPMENT OF THE CHATBOT

A chatbot was designed and developed for a Java programming course, called Systems Programming, offered at Universidad Carlos III de Madrid (UC3M), Spain. The course covers topics like object-oriented programming, testing, recursion, data structures and algorithms applied on them. This chatbot utilizes OpenAI technologies and the RAG technique to enhance the learning experience for students. This chatbot is offered as a service through a web application (see Figure 1) and supports conversations in English and Spanish. This chatbot functions similarly to other chatbots previously developed at UC3M under the CharlietBOT brand [34]. However, each chatbot operates as a separate instance, utilizing its own specific knowledge database for RAG, which aims to provide context-specific responses.

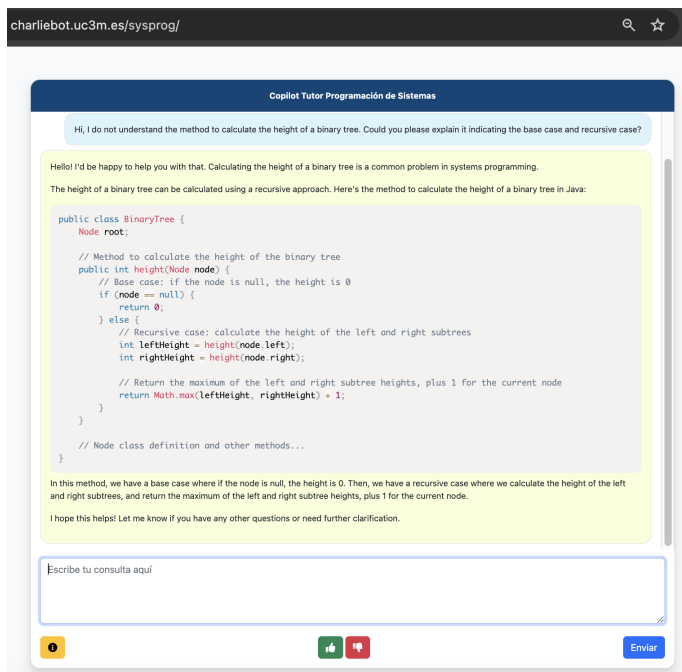


Fig. 1. A screenshot of the chatbot explaining the recursive method used to calculate the height of a binary tree. Students interact with the chatbot through a web application. They input their prompts in the text box at the bottom and submit them using the “Enviar” (send) button. Students can assess the response using thumbs-up and thumbs-down buttons to provide positive or negative feedback.

The context of the Systems Programming course was integrated into the chatbot using the RAG technique [18] and embeddings. These embeddings semantically represent the text and retrieve relevant information from a course-specific database, enhancing the chatbot’s accuracy and relevance to the material. The knowledge database was populated with content from MOOCs available on the edX platform, which had been previously developed by the course instructors at UC3M between 2015 and 2019 [20] (see Figure 2) both in English and Spanish. These MOOCs include videos, quizzes, programming exercises, and other supplementary resources. The subtitles for the videos were already available. The

text from PDF files was extracted using OpenAI’s Vision [35], and additional complementary materials, such as quizzes, programming exercises and slide presentations, were also added to the knowledge database. Text content analysis was performed using GPT-4, while embeddings were generated with OpenAI’s Ada embedding model [36], including weights for each embedding. All data were stored in a PostgreSQL database, chosen specifically for its compatibility with the pgvector extension [37], which is necessary to store the weight vectors associated with the Ada model embeddings.

Students interact with the chatbot via a web application. Prompts are processed using information from the knowledge database, prior student conversations, and predefined guidelines on the chatbot’s behavior. GPT-3.5-turbo with a 16K token window was chosen for its cost-benefit advantage over GPT-4. The architecture allows easy switching between LLMs for future updates. The main technological challenge in developing this chatbot was the limited token window (16K tokens in the GPT-3.5-turbo model), which required careful management of the conversation context to avoid losing important information. To overcome this, a dynamic pruning strategy [38] was implemented. This approach involves discarding older conversation context while retaining the most recent interaction, and crucially, storing relevant nouns in case the student refers to them with pronouns in subsequent interactions. This method ensures that the chatbot maintains coherent and contextually aware responses throughout the conversation, despite the token limit.

### IV. METHODOLOGY

This section outlines the methodology employed to carry out the experiment with the chatbot in the Systems Programming course. The following subsections provide details of the usage scenario and the instruments used for data collection.

#### A. Scenario

The Systems Programming course is offered during the second (spring) semester and is a compulsory subject for students enrolled in various telecommunication engineering degrees at UC3M. The chatbot was made available to students from Systems Programming starting from the second week of April (April 8, 2024) until the end of the academic year (May 13, 2024), providing students with over a month to use the chatbot freely, without the need for registration and with full anonymity on the messages sent to the chatbot. The course coordinator sent a message through the regular Learning Management System (LMS) to encourage students from all cohorts to use the chatbot, providing them with a link for easy access. A follow-up reminder was also sent through the LMS a few days later.

#### B. Instruments

Two instruments were used for data collection: logs of students’ conversations with the chatbot and a voluntary, anonymous survey completed by students at the end of the experiment. The logs captured a total of 1,059 messages

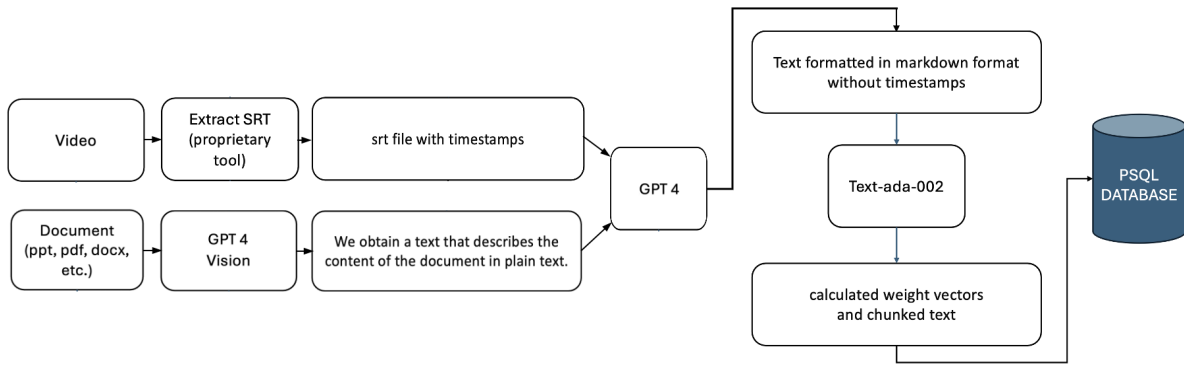


Fig. 2. Approach followed to populate the knowledge database (PostgreSQL - PSQL) for the chatbot.

exchanged between the students and the chatbot. In addition, 30 responses to the survey were collected from students who had used the chatbot during the experiment period.

1) *Conversation Logs*: These logs captured detailed records of student interactions with the chatbot, including both the prompts entered by students and the responses provided by the chatbot. Each entry in the data set includes unique identifiers for each conversation (organized as sessions in which the same browser tab was used), along with the total number of messages exchanged during the conversation. Additionally, each conversation was accompanied by qualitative rating reflecting student satisfaction (categorized as positive, negative, or neutral). By default, all conversations were marked as neutral unless the student actively rated them with a thumbs-up or thumbs-down in the chatbot web application. The logs provide detailed information for each conversation, including: 1) *User Role* (identifies whether the message was sent by the student or the chatbot.); 2) *Message Content* (contains the text of the student's prompt or the chatbot's response.); and 3) *Timestamp* (records the exact date and time when each message was sent). Figures 3 and 4 display the Dashboard developed for analyzing conversation logs, as well as an example of a conversation snippet.

2) *Survey*: The survey was structured to collect qualitative and quantitative data on students' experiences with the chatbot through 25 questions divided in four sections. The first section inquired whether the student had used the chatbot or not. If the answer was positive, students moved on to sections two through four, which focused on: a) the perceived usefulness of the chatbot, b) the usability of the chatbot, and c) the integration of the chatbot at the university level.

- a) *Perceived usefulness of the chatbot*: This section gathered information on how effectively the chatbot fulfills its educational purpose. The questions were designed to evaluate whether the chatbot aided in understanding and reviewing course material, and its usefulness in exam preparation and completing assignments. Students were specifically asked to rate the usefulness of the chatbot in explaining complex programming concepts, providing accurate answers, and improving their overall learning ex-

Copy	CSV	Excel	PDF	Print	Search: <input type="text"/>
Número	ID	Número de Mensajes	Valoración	Acciones	
86	nUwkaZX2rPz4Naw	153	Neutra	<a href="#">Ver</a>	
23	K4x6SLm6MU43OwS	141	Positiva	<a href="#">Ver</a>	
52	yvLiCCKZKealHnQ	132	Neutra	<a href="#">Ver</a>	
18	W6SBzwqa8Q28Hu9	123	Neutra	<a href="#">Ver</a>	
39	SJhU3bJr1603MnV	84	Positiva	<a href="#">Ver</a>	
11	GvIGTD0gquec5h0	78	Neutra	<a href="#">Ver</a>	
146	syldxFpcNKi52Zk	75	Neutra	<a href="#">Ver</a>	
47	T6m3Mhln4Sr0X60	71	Neutra	<a href="#">Ver</a>	
67	wwHz5JACag7QGgg	66	Neutra	<a href="#">Ver</a>	
43	YMOPmlomofmOI07	63	Neutra	<a href="#">Ver</a>	
109	8yuarZP8sdWtpxl	60	Positiva	<a href="#">Ver</a>	
92	vLXhWvj7387mtKF	60	Positiva	<a href="#">Ver</a>	
66	iJDjDIFR2j6hoJt	60	Neutra	<a href="#">Ver</a>	
58	XkBwmg2qOm00MHu	60	Neutra	<a href="#">Ver</a>	
36	9rwsAZXimlCRzWO	59	Neutra	<a href="#">Ver</a>	

Fig. 3. Screenshot of the Dashboard developed to collect and analyze conversation logs. The Dashboard organizes conversations by sessions, displaying the session number ("Número") and ID, the number of messages sent by the student ("Número de Mensajes"), the rating ("Valoración"), which can be positive, neutral or negative, and available actions ("Acciones") to view the content of the messages exchanged between the students and the chatbot.

perience in the course. Additionally, the survey assessed the ability of the chatbot to understand user queries and the relevance and accuracy of its responses, including any potential errors. Finally, students were asked if they felt that the responses of the chatbot were tailored specifically to the Systems Programming course.

- b) *Usability of the chatbot*: This section focused on the usability of the chatbot, including the web application through which students interacted with it. To assess usability, the System Usability Scale (SUS) [39], a widely recognized tool for measuring usability, was employed. Questions in this section asked students about the ease of interacting with the chatbot, the intuitiveness of its

Rol	Mensaje	Fecha
User	explain to me the current.getNext() method in nodes	08:30:43 - 06/05/2024
semantic_search		08:30:43 - 06/05/2024
CharlieBOT	<p>In the context of linked lists, the <code>getNext()</code> method is a common method used in node classes to retrieve the reference to the next node in the list. Here's an explanation of how the <code>getNext()</code> method works in the context of nodes:</p> <ol style="list-style-type: none"> <li><b>Node Class:</b> In a linked list, each element in the list is represented by a node. Each node contains two main components: <ul style="list-style-type: none"> <li>Data: The actual value or payload of the node.</li> <li>Reference to the Next Node: A reference or pointer to the next node in the list.</li> </ul> </li> <li><b><code>getNext()</code> Method:</b> The <code>getNext()</code> method is a simple accessor method that allows you to retrieve the reference to the next node in the list. It is typically defined in the node class and returns the reference to the next node.</li> </ol> <p>Here's an example of how the <code>getNext()</code> method might be implemented in a node class:</p> <pre> public class Node {     private int data;     private Node next;      // Constructor and other methods...      public Node getNext() {         return next; // Return the reference to the next node     } } </pre> <ol style="list-style-type: none"> <li><b>Usage:</b> When you have a reference to a node, you can use the <code>getNext()</code> method to access the next node in the list. For example: <pre> Node currentNode = // Obtain a reference to a node Node nextNode = currentNode.getNext(); // Retrieve the reference to the next node </pre> <p>In this example, <code>nextNode</code> will hold the reference to the next node in the list, allowing you to traverse the linked list by following the references from one node to the next.</p> <p>The <code>getNext()</code> method is fundamental for navigating through linked lists and performing operations such as traversal, insertion, and deletion.</p> <p>If you have any more questions related to systems programming or linked lists, feel free to ask!</p> </li> </ol>	08:30:50 - 06/05/2024

Fig. 4. Example conversation snippet between a student and the chatbot as displayed in the Dashboard, showing the role (“Rol”) (User of chatbot-CharlieBOT), message (“Mensaje”), and date (“Fecha”) for each iteration.

design, and the learning curve required to use it. This aimed to evaluate the user-friendliness of the chatbot and the ease with which students could engage with it.

- c) *Integration of the chatbot at the university level:* In this last section, students were invited to share their opinions on the integration of the chatbot with the institutional LMS, as well as their thoughts on the potential for implementing one or more chatbots in other courses within their engineering degrees.

## V. RESULTS

This section presents the key findings drawn from both the conversation logs and the survey responses collected from students that used the chatbot at the end of the experiment in the Systems Programming course.

### A. Conversation logs

The logs of the anonymous conversations between students and the chatbot provide insights into three key aspects: 1) the level of student activity, measured by the number of messages sent, 2) the response time of the chatbot, and 3) the length of both the messages sent by students and the responses generated by the chatbot.

1) *Student activity:* Figure 5 shows student activity with the chatbot, tracking messages sent between April 8, 2024 (the first day the chatbot became available) and May 13, 2024 (the last day of the experiment). In total, students sent 1,059 messages to the chatbot. The data reveal several peaks and troughs, reflecting fluctuating participation levels. Notably, the peak during the week of May 7–10 coincides with the continuous assessment exams, suggesting increased use of the chatbot for exam preparation. After May 10, the message volume drops sharply as the course had ended.

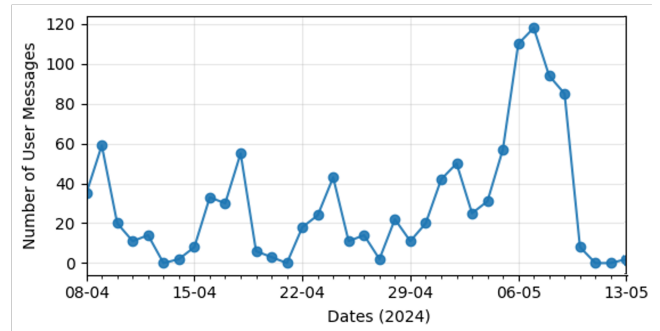


Fig. 5. The number of messages sent by students to the chatbot for each day during the experiment (April 8, 2024 – May 13, 2024). The data reveal a noticeable peak in the final days of the experiment, which coincides with the week of continuous assessment exams.

2) *Response time:* The RAG technique introduces some delay in response time due to the additional steps involved in retrieving relevant information before generating a response. Figure 6 shows the response time of the chatbot to student requests. The chatbot consistently delivers responses in under 5 seconds, which is reasonable from the student’s point of view. Only in a few cases the response times were longer, with the maximum recorded response time being 49 s. However, only 3.47% of the student interactions exceeded a response time of 10 seconds. Further analysis indicates that the longest response times occurred on days with the highest volume of messages.

3) *Message length:* Figure 7 illustrates the message length distribution, highlighting a clear contrast between the messages sent by the students and the responses provided by the chatbot. The responses provided by the chatbot are sig-



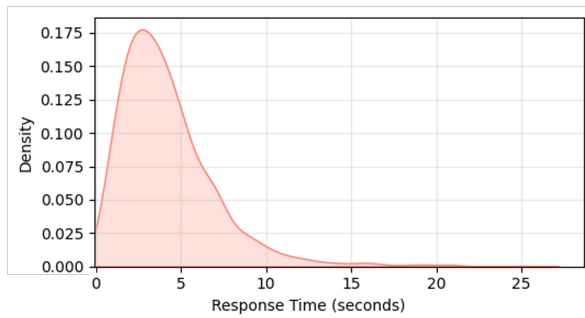


Fig. 6. The distribution of the chatbot's response time (in seconds)

nificantly longer, with an average length of 873 characters, reflecting its tendency to offer detailed and comprehensive explanations. In contrast, student messages are much shorter, averaging only 52.5 characters per message, indicating that students typically use concise prompts. Figure 4 provides an example of a very short message from a student followed by a long, detailed response from the chatbot. Although longer messages from students have been observed, these typically occur when students submit complete exercise statements or problems, expecting solutions from the chatbot. The length of student messages is a key indicator, as providing sufficient context within prompts is usually necessary when using general-purpose chatbots to ensure more accurate responses.

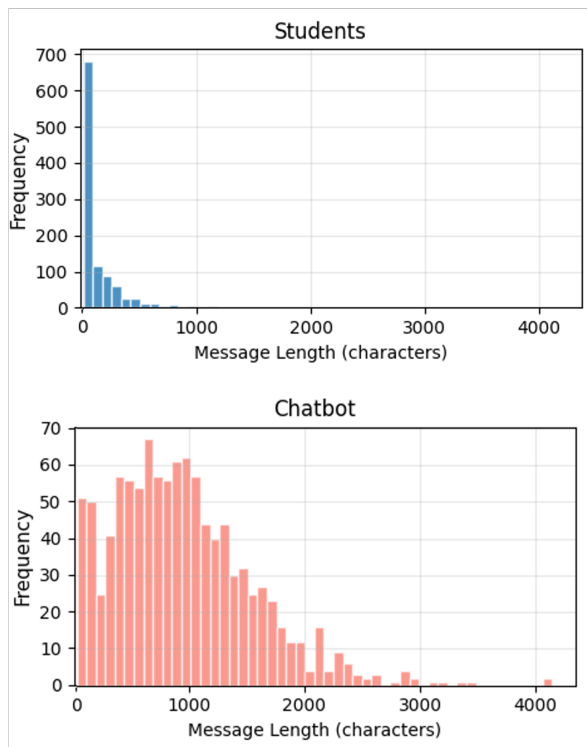


Fig. 7. The message length distribution is presented for both students and the chatbot. The top chart illustrates the distribution of message lengths sent by students, while the bottom chart displays the message length distribution of the chatbot's responses.

## B. Survey

A total of 30 students completed the survey indicating that they had used the chatbot. Of these, 8 students (26.7%) reported using it more than 10 times, 7 students (23.3%) used it between 6 and 10 times, 12 students (40%) used it between 3 and 5 times, and 3 students (10%) used it less than 3 times. Figure 8 shows the main reasons students used the chatbot, with the majority seeking to clarify theoretical concepts (26 students, 86.7%) and many using it to get an explanation on a piece of code (14 students, 46.7%).

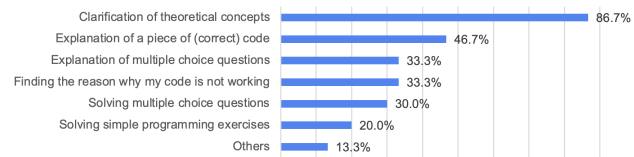


Fig. 8. Main reasons for students to use the chatbot (with each student able to select multiple answers). The most common reason was the clarification of theoretical concepts, which was selected by the majority of students.

1) *Perceived usefulness of the chatbot:* Students gave positive feedback on the usefulness of the chatbot developed using the RAG technique. A total of 83.3% of students (25) reported that the chatbot always or almost always correctly understood their questions, while 90% of students (27) stated that the chatbot's answers were always or almost always clear and understandable. Notably, 93.3% of students (28) felt that the chatbot's responses were specifically tailored to the Systems Programming course, which is due to the use of the RAG technique in the development of the chatbot. Furthermore, on a scale from 1 (minimum) to 5 (maximum), 86.7% of students (26) rated the chatbot's usefulness in improving their learning experience with a score of 4 (50%, 15 students) or 5 (36.7%, 11 students). Additionally, 73.3% of students (22) rated the chatbot's usefulness in assisting their study efforts with a score of 4 (36.7%, 11 students) or 5 (36.7%, 11 students).

It is also interesting to examine how students reacted to the answers provided by the chatbot. A total of 43.3% of students (13) cross-checked the chatbot's responses with course materials to verify their accuracy, while 36.7% of students (11) directly used the solutions provided by the chatbot without confirming their correctness. Other, less common reactions included using Google to verify the chatbot's answers, attempting to understand the code provided by the chatbot, comparing it with solutions from other AI-powered chatbots, and testing whether the code actually worked.

Regarding potential errors in the responses provided by the chatbot, 53.3% of students (16) reported finding no errors. The remaining students encountered issues such as the chatbot failing to answer the specific question, providing incorrect explanations, or offering code that compiled and ran but did not perform the intended task. One notable case involved a student asking about an outdated topic no longer covered in the Systems Programming course, to which the chatbot could not provide a response; this is something to be expected as the knowledge database did not include that content.

2) *Usability of the chatbot:* The usability of the chatbot was evaluated using the System Usability Scale (SUS) [39], a widely recognized tool to measure usability across software applications, websites, and similar platforms. The SUS consists of 10 items, with responses ranging from Strongly Agree to Strongly Disagree, alternating between positive (odd-numbered) and negative (even-numbered) statements. After calculating the SUS scores for each of the 30 students who completed the survey, a score of 72.25 was obtained. According to SUS interpretation guidelines, this score is classified as “Good” and corresponds to a grade of B, indicating that the usability of the chatbot used in Systems Programming is above average.

3) *Integration of the chatbot at the university level:* Students were asked whether they would like the chatbot to be integrated within the institutional LMS (Moodle, in this case), making the chatbot part of the same ecosystem they use to access course materials, complete exercises, submit assignments, and more. All students (100%) supported the idea of integrating the chatbot into the institutional LMS. In addition, students were asked if they would find it useful to have personalized chatbots for each of the courses they are enrolled in. Of the respondents, 90% of the students (27) expressed interest in having course-specific chatbots, while the remaining 10% indicated that their preference would depend on the course. There were no negative responses to this question. Lastly, students were asked whether they would prefer a single chatbot for all courses or a dedicated chatbot for each course. The majority (60%) preferred having a chatbot per course, while 40% preferred a single chatbot for all courses. This aligns with the notion that chatbots tailored to the specific context of each course can provide more effective support for students.

## VI. CONCLUSIONS AND FUTURE WORK

This article has presented a chatbot specifically developed for a Java programming course called Systems Programming, and an experiment of the use of this chatbot for more than one month in April-May 2024. This chatbot was developed using technologies provided by OpenAI (GPT-3.5-turbo and GPT-4), RAG, embeddings, and a knowledge base built with content that was already available for this course (content from videos, quizzes, programming exercises, and supplementary material). The evaluation of the chatbot by the students was positive both in terms of the usefulness and usability of the chatbot; students also highlighted their perception that the chatbot appeared to be tailored to the specific context of the Systems Programming course.

Communication with GenAI-powered chatbots requires specific skills, particularly in creating effective prompts (a practice often referred to as “Prompt Engineering” [40]). Although some research suggests that “Prompt Engineering” could eventually be optimized by GenAI models themselves [41], [42], it remains crucial to provide sufficient context during chatbot conversations to improve the quality of responses. Some students may not yet possess these skills (especially

in the case of first-year students). Therefore, it is important to support students in their interactions with chatbots, for example, by offering chatbots that already have a contextual understanding of the course.

Ideally, each course could have its own dedicated chatbot, tailored to the specific course context. Such chatbots could be integrated in the institutional LLM, and leverage the RAG technique and a knowledge base populated with course-specific content. This would provide students with a 24/7 AI tutor to complement the support they receive from their instructors. The potential of this approach has been demonstrated in this article.

This article opens several avenues for future research and development. First, it would be valuable to test different LLMs to generate responses to student queries and evaluate student satisfaction with these responses; currently, the chatbot uses GPT-3.5-turbo, but other LLMs could be tested, with the option to update the chatbot as new models become available, while assessing the cost implications of each. Second, the chatbot could be expanded to support students with more specific tasks, such as lab assignments or course projects. Third, student messages could be analyzed in detail to identify frequently asked questions, while chatbot responses could be reviewed to detect inaccuracies; the impact of incorporating Chain-of-Thought prompting [43] could also be explored to enhance the quality of responses. Fourth, the knowledge database could be further enriched with additional content, including multimodal resources, to improve the chatbot’s contextual awareness. Fifth, this approach could be extended to develop similar chatbots for other courses with rich materials, leveraging the RAG technique while utilizing the same underlying structure. Sixth, a more in-depth study could be conducted comparing general-purpose chatbots with domain-specific chatbots that make use of the RAG technique, assessing their effectiveness in providing accurate and context-relevant academic support. Finally, a further study could be conducted to compare the effectiveness of general prompts versus enhanced or specific prompts in improving the ability of the chatbot to deliver precise, context-aware responses.

## ACKNOWLEDGMENTS

This work was supported by grants PID2023-146692OB-C31 (GENIE Learn project) and PID2020-112584RB-C31 (H2O Learn project) funded by MICIU / AEI / 10.13039 / 501100011033, MCIN / AEI / 10.13039 / 501100011033 and ERDF, EU. This research has also received partial support from the European Union through Erasmus+ projects MICROCASA (101081924 ERASMUS-EDU-2022-CBHE-STRAND-2), MICRO-GEAR (101127144 ERASMUS-EDU-2023-CBHE-STRAND-3), POEM-SET (2021-FR01-KA220-HED-000032171) and EcoCredGT (101129122 ERASMUS-EDU-2023-CB-VET). Views and opinions expressed are, however, those of the authors only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

## REFERENCES

- [1] A. Pardo, J. Jovanovic, S. Dawson, D. Gašević, and N. Mirriahi, "Using learning analytics to scale the provision of personalised feedback," *British Journal of Educational Technology*, vol. 50, no. 1, pp. 128-138, 2019.
- [2] H. S. Nwana, "Intelligent tutoring systems: an overview," *Artificial Intelligence Review*, vol. 4, no. 4, pp. 251-277, 1990.
- [3] E. Mousavinasab, N. Zarifasanaiey, S. R. Niakan Kalhori, M. Rakhshan, L. Keikha, and M. Ghazi Saeedi, "Intelligent tutoring systems: a systematic review of characteristics, applications, and evaluation methods," *Interactive Learning Environments*, vol. 29, no. 1, pp. 142-163, 2021.
- [4] M. W. Ashfaq, S. Tharewal, S. Iqbal, and C. N. Kaye, "A Review on Techniques, Characteristics and Approaches of an Intelligent Tutoring Chatbot System," in *2020 International Conference on Smart Innovations in Design, Environment, Management, Planning and Computing (ICSIDEMPC)*, pp. 258-262, 2020.
- [5] N. González-Castro, P. J. Muñoz-Merino, C. Alario-Hoyos, and C. Delgado Kloos, "Adaptive learning module for a conversational agent to support MOOC learners," *Australasian Journal of Educational Technology*, vol. 37, no. 2, pp. 24-44, 2021.
- [6] M. A. Kuhail, N. Alturki, S. Alramlawi, and K. Alhejori, "Interacting with educational chatbots: A systematic review," *Education and Information Technologies*, vol. 28, no. 1, pp. 973-1018, 2023.
- [7] G.-J. Hwang and C.-Y. Chang, "A Review of Opportunities and Challenges of Chatbots in Education," *Interactive Learning Environments*, vol. 31, no. 7, pp. 4099-4112, 2023.
- [8] C. Catalan Aguirre, N. Gonzalez Castro, C. Delgado Kloos, C. Alario-Hoyos, and P. J. Muñoz Merino, "Conversational agent for supporting learners on a MOOC on programming with Java," *Computer Science and Information Systems*, vol. 18, no. 4, pp. 1271-1286, 2021.
- [9] D. Baidoo-Anu and L. O. Ansah, "Education in the era of generative artificial intelligence (AI): Understanding the potential benefits of ChatGPT in promoting teaching and learning," *Journal of AI*, vol. 7, no. 1, pp. 52-62, 2023.
- [10] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. et al., "A survey on evaluation of large language models," *ACM Transactions on Intelligent Systems and Technology*, vol. 15, no. 3, pp. 1-45, 2024.
- [11] ChatGPT, Accessed: May 20, 2024. [Online]. Available: <https://chatgpt.com/>
- [12] Microsoft Copilot, Accessed: May 20, 2024. [Online]. Available: <https://copilot.microsoft.com/>
- [13] Google Gemini, Accessed: May 20, 2024. [Online]. Available: <https://gemini.google.com/>
- [14] Claude, Accessed: May 20, 2024. [Online]. Available: <https://claude.ai/>
- [15] A. Strzelecki, "To use or not to use ChatGPT in higher education? A study of students' acceptance and use of technology," *Interactive Learning Environments*, pp. 1-14, 2023.
- [16] M. Firat, "What ChatGPT means for universities: Perceptions of scholars and students," *Journal of Applied Learning and Teaching*, vol. 6, no. 1, pp. 57-63, 2023.
- [17] E. Theophilou, C. Koyutürk, M. Yavari, S. Bursic, G. Donabauer, A. Telari, A. Testa, et al., "Learning to Prompt in the Classroom to Understand AI Limits: A pilot study," in *International Conference of the Italian Association for Artificial Intelligence*, Cham: Springer Nature Switzerland, pp. 481-496, 2023.
- [18] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459-9474, 2020.
- [19] J. Chen, H. Lin, X. Han, and L. Sun, "Benchmarking large language models in retrieval-augmented generation," in *Proc. of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 16, pp. 17754-17762, 2024.
- [20] C. Alario-Hoyos, I. M. Estévez Ayres, C. Delgado Kloos, P. J. Muñoz Merino, E. Llorente Pérez, and J. Villena Román, "Redesigning a freshman engineering course to promote active learning by flipping the classroom through the reuse of MOOCs," *International Journal of Engineering Education*, vol. 35, no. 1, pp. 385-396, 2019.
- [21] J. Weizenbaum, "ELIZA—a computer program for the study of natural language communication between man and machine," *Communications of the ACM*, vol. 9, no. 1, pp. 36-45, 1966.
- [22] E. Adamopoulou and L. Moussiades, "Chatbots: History, technology, and applications," *Machine Learning with Applications*, vol. 2, Art. no. 100006, pp. 1-18, 2020.
- [23] N. A. Godse, S. Deodhar, S. Raut, and P. Jagdale, "Implementation of chatbot for ITSM application using IBM Watson," in *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pp. 1-5, 2018. IEEE.
- [24] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, pp. 1-11, 2017.
- [25] K. Ramesh, S. Ravishankaran, A. Joshi, and K. Chandrasekaran, "A survey of design techniques for conversational agents," in *International Conference on Information, Communication and Computing Technology*, pp. 336-350, Springer, 2017.
- [26] S. Hussain, O. A. Sianaki, and N. Ababneh, "A survey on conversational agents/chatbots classification and design techniques," in *Web, Artificial Intelligence and Network Applications: Proc. Workshops of the 33rd International Conference on Advanced Information Networking and Applications (WAINA-2019)*, vol. 33, pp. 946-956, Springer, 2019.
- [27] A. Stojanov, Q. Liu, and J. H. L. Koh, "University Students' Self-Reported Reliance on ChatGPT for Learning: a Latent Profile Analysis," *Computers and Education: Artificial Intelligence (in press)*, vol. 100243, 2024.
- [28] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "QLoRA: Efficient finetuning of quantized LLMs," in *Advances in Neural Information Processing Systems*, 2024.
- [29] R. Liu, C. Zenke, C. Liu, A. Holmes, P. Thornton, and D. J. Malan, "Teaching CS50 with AI: leveraging generative artificial intelligence in computer science education," in *Proc. 55th ACM Technical Symposium on Computer Science Education V. 1*, pp. 750-756, ACM, 2024.
- [30] Khanmigo, Accessed: May 20, 2024. [Online]. Available: <https://khanmigo.ai/>
- [31] E. Frankford, C. Sauerwein, P. Bassner, S. Krusche, and R. Breu, "AI-Tutoring in Software Engineering Education," *arXiv preprint arXiv:2404.02548*, 2024.
- [32] H. Jin, S. Lee, H. Shin, and J. Kim, "Teach AI How to Code: Using Large Language Models as Teachable Agents for Programming Education," in *Proc. CHI Conference on Human Factors in Computing Systems*, pp. 1-28, 2024.
- [33] GPTs, OpenAI, Accessed: May 20, 2024. [Online]. Available: <https://chat.openai.com/gpts>
- [34] "CharlieBot: Improving Digital Communication with Artificial Generative Intelligence," Universidad Carlos III de Madrid, 2023. Accessed: May 20, 2024. [Online]. Available: <https://uc3m.es/sdic/articles/2023/charliebot-improving-digital-communication-with-artificial-generative-intelligence>
- [35] Vision, Accessed: May 20, 2024. [Online]. Available: <https://platform.openai.com/docs/guides/vision>
- [36] Embedding models, Accessed: May 20, 2024. [Online]. Available: <https://platform.openai.com/docs/guides/embeddings/embedding-models>
- [37] pgvector. PostgreSQL: pgvector 0.5.0, Accessed: May 20, 2024. [Online]. Available: <https://postgresql.org/about/news/pgvector-050-released-2700/>
- [38] X. Ma, G. Fang, and X. Wang, "LLM-Pruner: On the structural pruning of large language models," *Advances in Neural Information Processing Systems*, vol. 36, pp. 21702-21720, 2023.
- [39] A. Bangor, P. T. Kortum, and J. T. Miller, "An empirical evaluation of the system usability scale," *International Journal of Human-Computer Interaction*, vol. 24, no. 6, pp. 574-594, 2008.
- [40] P. Denny, V. Kumar, and N. Giacaman, "Conversing with copilot: Exploring prompt engineering for solving cs1 problems using natural language," in *Proc. 54th ACM Technical Symposium on Computer Science Education V. 1*, pp. 1136-1142, 2023.
- [41] D. Genkinay, "AI Prompt Engineering Is Dead · Long live AI prompt engineering," *IEEE Spectrum*, 2024. [Online]. Available: <https://spectrum.ieee.org/prompt-engineering-is-dead>
- [42] R. Battle and T. Gollapudi, "The unreasonable effectiveness of eccentric automatic prompts," *arXiv preprint arXiv:2402.10949*, 2024.
- [43] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, et al., "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24824-24837, 2022.