JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering, Vol. 18, No. 2, June 2024, pp. 125-133

Academic Query Assistant: Integrating LLM API into an Academic Assistant Using a Microservices Architecture

Pedro Fernando Álvarez* **and Sebastian Quevedo**

Unidad Académica de Informática, Ciencias de la Computación, e Innovación Tecnológica, Grupo de Investigación Simulación, Modelado, Análisis y Accesibilidad (SMA²), Universidad Católica de Cuenca, Cuenca, Ecuador

pedro.alvarez.22@est.@ucacue.edu.ec, asquevedos@ucacue.edu.ec

Abstract

Artificial intelligence (AI) has made impressive progress in recent years. One notable development in this technology has been the emergence of large language models (LLMs) that are capable of generating and interpreting natural language data. These models have gained widespread attention for their remarkable text generation capabilities and improved user interface. At present, academic institutions face challenges associated with how to access vast amounts of information in an efficient manner. This problem is compounded by the increasing number of academic documents available, the dispersion of information in different repositories, and the time and resources required to search and filter this information, which represents a significant workload for professors and students. To address the issue, the current paper proposes an AI-powered assistant integrated with LLMs and a software system based on a microservices architecture. This assistant offers clear and contextually relevant answers to help make academic information retrieval processes more efficient. Altogether, this article proposes an AI-powered assistant that covers the integration aspects of both AI and software models. It also uses intelligent assistants to manage academic information, and is intended to serve as a model for future implementations.

Category: Natural Language Processing

Keywords: Assistant; API; LLM; Knowledge retrieval; NLP

I. INTRODUCTION

Recent advancements in artificial intelligence (AI) have marked the beginning of a new digital era [1-4]. Specific types of AI models, which are known as large language models (LLMs), have stood out due to their incredible capabilities in generating and interpreting natural language data [5].

LLMs can develop significant amounts of new text based on small input requests [6]. The broad public release of ChatGPT by OpenAI in November 2022 marked a substantial increase in the fundamental ability of the software to create new text using refined models (GPT-3.5), along with an improved user interface [7]. This release led to a substantial increase in the public discourse about how LLMs can impact educational integrity [6].

Advanced language models such as LLMs have the potential to provide professional assistance in various fields, including education [8]. They offer a range of benefits, such as enhancing user interactions with digital platforms, particularly in areas that are related to customer service. These language models have sophisticated capabilities to improve the user experience by providing efficient and effective solutions.

Open Access http://dx.doi.org/10.5626/JCSE.2024.18.2.125 http://jcse.kiise.org

Received 26 April 2024; **Accepted** 10 June 2024

*Corresponding Author

pISSN: 1976-4677 eISSN: 2093-8020

According to [9], using AI, especially LLMs, in software engineering education is not just a matter of following a trend but a necessity. The introduction of LLMs has posed crucial questions about their impact on the process and implementation of software engineering. The AI-powered assistant presented herein is geared toward simplifying the task of retrieving academic information. It is anticipated that this technology is expected to contribute to the productivity of academic staff.

Application programming interface (API) integration holds significant implications for developers seeking to harness the power of external services and functionalities within their serverless applications. As explored in a prior study [10], the effectiveness of API integration mechanisms is paramount in enabling developers to leverage a wide range of services and data sources in their serverless workflows.

It is important for educational institutions to have an efficient system that ensures quick access to information and prompt responses to any queries that are related to administrative and academic documents. This problem can be solved by AI-powered assistants making use of technology and automation. This can facilitate document management as well as enable students and administrative staff to access information smoothly. However, it can be difficult to find immediate information due to the large amounts of academic and administrative documents available. To overcome this challenge, the present study developed a modern LLM-based assistant to leverage documents as a single source of information to respond to academic staff queries effectively. Institutions store vast amounts of valuable information in their documents, which contain relevant details that are related to their functioning. However, it can be difficult for students and administrative staff to access this information due to the scattered documentation, which leads to long wait times.

The assistant presented herein is powered by OpenAI's advanced natural language processing (NLP) models. Its purpose is to provide quick and accurate answers to user queries that are related to accessing academic information, thus eliminating the need for manual scanning of multiple documents. This project ultimately aims to solve the problem of long wait times for students and staff when seeking answers to their questions or concerns.

The article presents the comprehensive process involved in developing an innovative assistant that is integrated with AI technology and microservices software technology. The rest of this paper covers the Related Work, Methodology, Results and Discussion, and Conclusions and Future Work.

II. RELATED WORK

Numerous studies in the field of AI have explored various professional domains, such as medicine, education, and public administration. These studies have mostly investigated the potential utility of various modern technologies in these fields. For example, studies have shown that AI can assist with image interpretation [11] and the segmentation of tumors [12] in the medical field. Similarly, in public administration, question-answering systems are being developed with the help of AI [2]. Meanwhile, in the field of education, AI-powered tools such as ChatGPT [3] are being used to support students in decision-making [4].

In [1], the author discusses the potential of using an OpenAI chatbot to transform the medical fields, including in terms of diagnostics, treatment planning, and healthcare delivery, thus complementing the idea of increased performance in professional and administrative fields.

Duy and Thanh [2] describe the development of a question-answering (QA) system designed for public administrative services in Vietnam. The focus of this work is on building a legal QA system that provides answers which are related to passages in law documents. Retrieval models are used for query handling related to documents. This assistant is not only designed to use files as a source of information, but it can also offer students with tailored guidance based on their interests.

Mhlanga [3] emphasizes the potential use of LLMs such as ChatGPT in revolutionizing the educational sector. One key advantage of this research is its accessibility, as it is capable of handling multiple languages, thus making it more accessible to people all over the world.

Abu-Rasheed et al. [4] propose a chatbot that integrates LLMs to mediate between students while assisting them in decision-making during the learning process. This study exhibits positive feedback regarding the Chatbot's functionality. It aims to leverage the latest advancements in AI, particularly the development of LLMs, to optimize communication, simplify processes, and enhance support in professional and educational settings. This study also intends to demonstrate the potential of AI-driven assistants to boost human capabilities.

In [11], the authors describe the development of a software system that uses a microservice architecture to ensure scalability, maintainability, and efficient AI model integration. This paper describes how microservices facilitate the easier deployment and management of assistant components, as each service can be developed, deployed, and scaled independently.

The limitations highlighted in prior studies [1-4, 10] emphasize the need for improvement in various areas related to AI-powered academic query management systems, such as the reliance on virtual machines, the lack of direct engagement with individual learners' needs, heavy dependency on language-specific models, and potential ethical concerns like bias and privacy breaches.

Incorporating these insights into academic query management systems can achieve significant improvements in various aspects of the proposed assistant. For instance, addressing the reliance on previously assigned virtual machines can ensure efficient query handling, thus avoiding overloads and slow startups [10]. Similarly, by considering the distinctive needs of individual users and avoiding over-reliance on language-specific models, Chatbots can offer users from diverse backgrounds more personalized and accessible support [2, 3].

Moreover, integrating ethical considerations and safeguards, such as bias mitigation techniques and privacy protection measures, can ensure that responsible and trustworthy assistance is delivered [1]. Further, the weaknesses identified in the prior study [4] suggest possible directions for future research to expand sample sizes, incorporate multiple LLMs, and integrate user-profile data, all of which can enhance the effectiveness and reliability of AI-driven assistants in supporting academic query management.

Considering these limitations and incorporating the insights from the studies is expected to make academic query management systems more robust, reliable, and ethical. This can improve the academic experience for both students and staff.

III. METHODOLOGY

This section presents a detailed overview of the proposed architecture, including the effective prompting engineering techniques used, the key aspects involved in creating an assistant, compatible files for knowledge retrieval, and user interface design. This document has been structured into subsections, each focusing on a specific aspect of the assistant.

A. Overview of the Proposed Assistant

Fig. 1 presents a visual overview of the proposed assistant. It goes through different stages to obtain a final response. We will thoroughly explore each step of creation and implementation, all the while demonstrating the efficacy of LLMs in retrieving information and document management.

- Front-End: Displays the answers issued by the assistant and serves as an intermediary for sending queries from the user.
- Back-End: Manages the existence of threads and is responsible for retrieving a response once the request has been processed.
- External Services: Processes user queries through the assistant that works with instructions to send answers.

B. External Services

Practical prompt engineering is essential in developing an intelligent assistant. It aims to furnish precise and comprehensive instructions that allow the AI model to produce relevant and reliable responses that meet the user's expectations. A skillfully crafted prompt can significantly enhance the accuracy and thoroughness of the answer given by the model, thus ensuring the user's satisfaction. This study employed various techniques to optimize interactions with the AI model. The instructions that the assistant follows to solve any academic query are shown in Fig. 2.

1) Define a Goal and Audience

The goal will determine the structure of the prompt to be designed in the following step and assist in evaluating the quality of the system's response before further iterations. A target audience is also clearly defined and described. Defining the prompt in the function of the audience makes it possible to adjust the tone, complexity, detail, and content to be provided by the assistant [13, 14]. An example is presented in Sections A and D in Fig. 2.

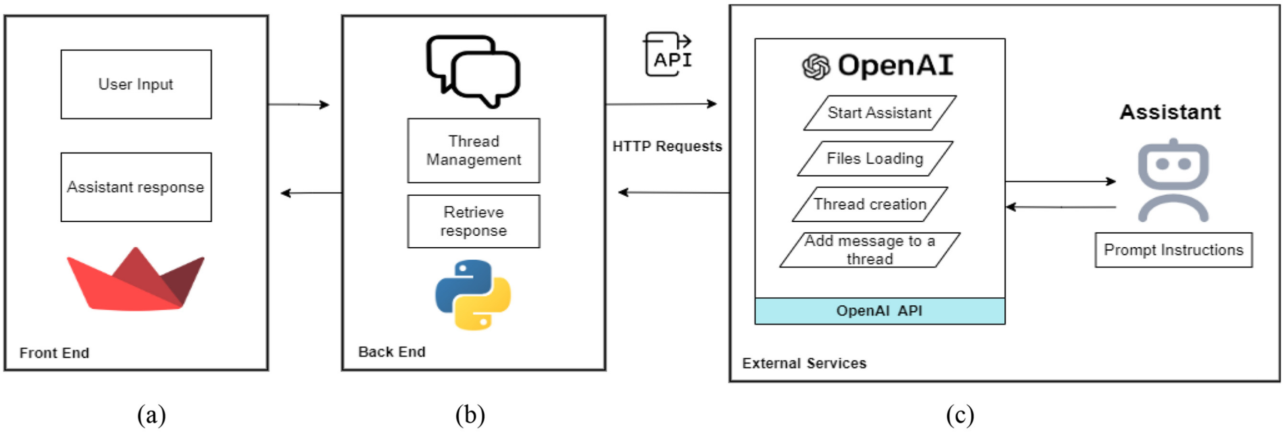
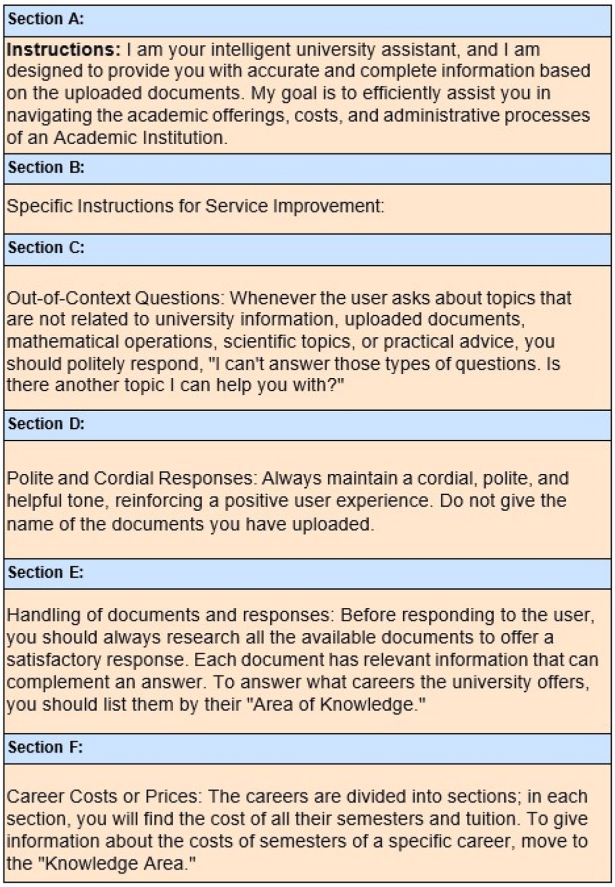


Fig. 1. Architecture of proposed AI-integrated assistant: (a) front-end in box, (b) back-end in box, and (c) external services in box.



Section A:

Instructions: I am your intelligent university assistant, and I am designed to provide you with accurate and complete information based on the uploaded documents. My goal is to efficiently assist you in navigating the academic offerings, costs, and administrative processes of an Academic Institution.

Section B:

Specific Instructions for Service Improvement:

Section C:

Out-of-Context Questions: Whenever the user asks about topics that are not related to university information, uploaded documents, mathematical operations, scientific topics, or practical advice, you should politely respond, "I can't answer those types of questions. Is there another topic I can help you with?"

Section D:

Polite and Cordial Responses: Always maintain a cordial, polite, and helpful tone, reinforcing a positive user experience. Do not give the name of the documents you have uploaded.

Section E:

Handling of documents and responses: Before responding to the user, you should always research all the available documents to offer a satisfactory response. Each document has relevant information that can complement an answer. To answer what careers the university offers, you should list them by their "Area of Knowledge."

Section F:

Career Costs or Prices: The careers are divided into sections; in each section, you will find the cost of all their semesters and tuition. To give information about the costs of semesters of a specific career, move to the "Knowledge Area."

Fig. 2. These instructions segmented into sections are used to smoothly create and operate the assistant. Each of these sections plays an essential role in understanding every single query.

2) Context as a Guide

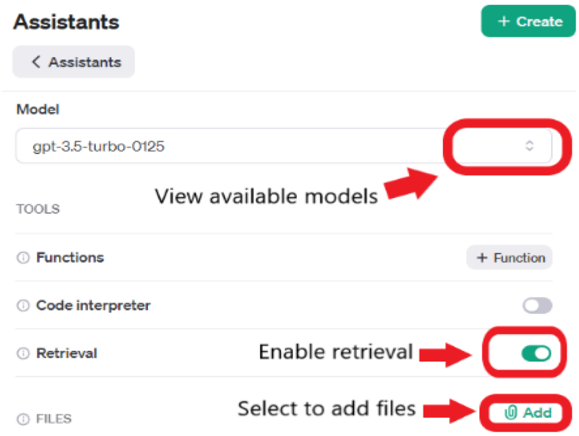
It is essential to provide context to help the assistant comprehend the situation. However, more information does not mean a better quality of response; it is necessary to omit data that does not contribute significantly to the understanding of the context. Providing a specific role to the assistant ensures that its responses are aligned with the desired outcome [15, 16]. An example can be seen in Sections E and F in Fig. 2.

3) Clear Instructions

Providing clear and descriptive instructions is an essential aspect of effectively directing the course of AI. This includes specifying whether one is seeking more creative or precision-focused responses, which will help the AI understand and execute an individual's objectives. Providing a precise and detailed prompt is crucial in generating content that is more aligned with the unique requirements of a given scenario [15, 16].

4) Evaluate and Adapt

Resampling is a continuous process that is based on the evaluation of the obtained output. Iteration is beneficial for improving instructions, making it possible to adjust the limitations or areas where the model presents problems, and to obtain multiples outputs to select the best one [16].



Model: gpt-3.5-turbo-0125

View available models

Enable retrieval

Select to add files

Fig. 3. A list of available models, how to enable the retrieval tool, and how to add files to the assistant.

5) Validation of Unnecessary Prompts

It is essential to give concrete instructions to ensure that the assistant stays focused and does not give unnecessary answers, thereby providing accurate and helpful answers, as shown in Sections B and C in Fig. 2. It is also essential to ensure that the files or information provided to the assistant are related to the query at hand, as this will help consolidate the process and avoid any confusion or delays.

6) Files Loading

Files can be uploaded easily from the development platform. These files are required to use the "knowledge retrieval" tool, which retrieves specific data from the uploaded files. As shown in Fig. 3, the retrieval tool can be enabled through the OpenAI developer platform.

7) Files Compatibility

The effectiveness and quality of assistant responses are directly related to the compatibility of file formats and their content. Therefore, an important aspect to consider here is the simplicity and commonality of formats that the assistant can manage to ensure that it comprehends them accurately.

In this aspect, specific attributes in the content of widely used PDF files can affect the assistant's ability to process retrieved information. It can be particularly difficult for the assistant to understand complex elements such as images, diagrams, and tables, which can potentially result in unclear or incorrect responses.

These files should be implemented in a clear and

Table 1. Available files for knowledge retrieval

Category	Format
Text documents	.txt, .md, .docx, .html, .json, .py, .rb, .java, .pdf, .tex
Presentation file	.pptx
Code file	.cpp, .c, .php, .js, .css, .ts

straightforward structure to ensure a precise interpretation and a relevant response. This approach is critical for achieving optimal results.

According to data provided by OpenAI [17], supported documents are presented in Table 1 [18-21].

C. Back-End Development

The OpenAI development platform is essential for creating an efficient AI-powered assistant. It features a range of tools designed to support specific needs. Users can write detailed instructions to support smooth task execution. The platform also presents a selection of models to find the best fit for the system's requirements.

The "Retrieval" tool allows the assistant to retrieve knowledge outside its NLP model through user-supplied documents. Once files are uploaded, OpenAI will automatically fragment the document, index and store the embeddings, and implement vector search to retrieve relevant information and answer the user's queries [17].

A list of available models can be seen and tested through assistant configuration, which is depicted in Fig. 3.

1) LLM Pricing and Models

NLP offers many possibilities for LLMs. There are various types of LLMs available, each with unique features that can be leveraged for NLP tasks. Table 2 presents a comparative analysis of the pricing of the most widely used LLMs to understand their capabilities relative to their cost.

Table 2 presents the cost per 1 million tokens for input and output prompts. There is an apparent reason why the "GPT-3.5-turbo" model was chosen, with cost being one of the most significant factors compared to the other available models. It offers a cost-effective option for NLP, and the quality of the response is not significantly compromised.

Table 2. LLM model prices

Model	Input	Output
GPT-3.5-turbo [18]	0.50	1.50
GPT-4-turbo [18]	10.00	30.00
Mistral Large [19]	8.00	24.00
Mistral Small [19]	2.00	6.00
Gemini Pro [20]	0.125	0.375
Claude Instant [21]	0.80	2.4
Claude 2.1 [21]	8.00	24.00

The assistant model was selected based on an evaluation of the implementation costs, particularly in academic settings where resources are often limited. The chosen model delivers a satisfactory balance between performance and price.

2) Thread Management

Messages are assigned to a thread representing the conversation session between the assistant and the user. There is no limit to the number of messages that the thread can store [17]. Threads serve as a record of all the messages that have been exchanged during a session, and they provide important context for future queries within that session. The creation of the thread is depicted in Fig. 4.

3) Query Processing

Once the query is sent, the assistant processes it using the "GPT-3.5-turbo" model and then enters an execution process representing different response stages. According to the data provided by OpenAI [17], the stages are presented in Table 3.

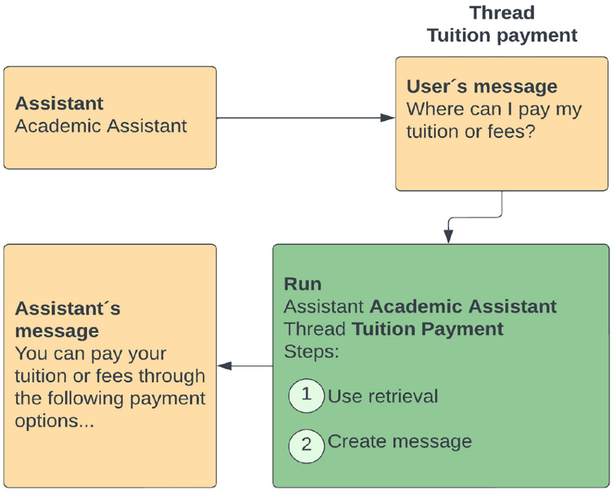


Fig. 4. Process from thread creation to response retrieval.

Table 3. Run steps and concepts

Category	Format
in progress	Assistant is analyzing user prompt
completed	The request was fulfilled
failed	Internal error
canceled	The request was canceled
expired	The request took too long to respond

A. Front-End Development

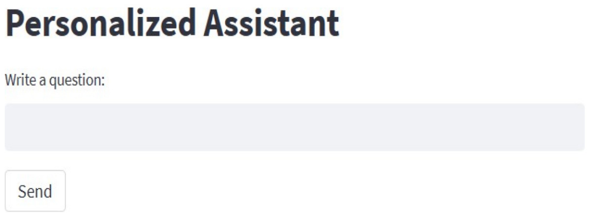
To interact with the OpenAI API through HTTP requests, a unique "API KEY" is required for authorization to access all functions. In light of this, OpenAI offers an application programming interface called the "Assistants API" that allows developers to build powerful intelligent assistants within their applications. This API eliminates the need to manage conversation history and also provides access to multiple tools, including "Code Interpreter," "Retrieval," and "Function Calling" [22].

1) Integration to Streamlit App

Integrating OpenAI services for assistant creation requires several parameters to enable interactions between the user and the API. It is essential to have a unique API key and an available assistant identifier, which are declared in the code to authorize access to OpenAI services. Once these parameters are configured, the API can be called from the application.

2) User Input Queries

The assistant presents a simple, easy-to-understand interface that allows users to perform academic queries in an efficient manner. Streamlit was used for its implementation; this framework is an open-source Python library that facilitates the creation of customized web applications for machine learning and data science [23]. When the initial screen is displayed, the interface has a text field where users can type their queries. A button to send queries is next to the input field, and pressing this button starts the communication process with the OpenAI assistant, as shown in Fig. 5.


Personalized Assistant

Write a question:

Fig. 5. Assistant interface developed through Streamlit.

3) Message Annotations

Assistants create messages with annotations that are embedded in the content array of the object. These annotations provide information on how to annotate the text in the message [17]. There is one type of annotation on this assistant:

- File citation: File citation annotations refer to a specific quote in a particular file that is uploaded and used by the assistant to generate the response.

IV. RESULTS AND DISCUSSION

The proposed AI-powered assistant for academic information retrieval is illustrated in Fig. 6, which showcases a historical chat session where the assistant followed a set of features designed to streamline the access and management of academic documents. This system offers robust functionalities to facilitate efficient access to relevant information within document retrieval.

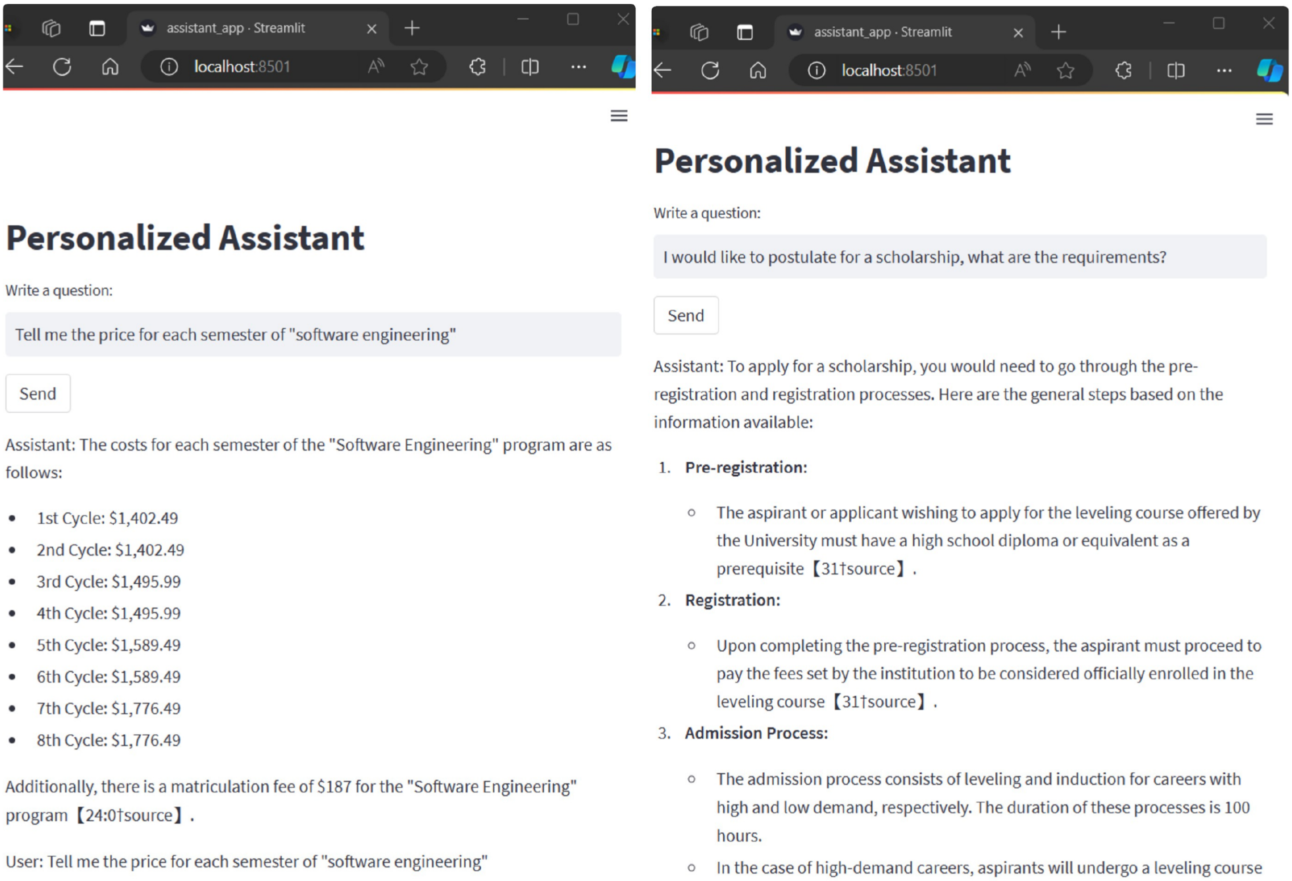
A series of comprehensive assessments were conducted to evaluate the AI-powered assistant's efficiency in comprehending a broad range of queries that is related to academic and administrative documents. The results, as presented in Fig. 6, demonstrate that the assistant can provide exact responses that are closely aligned with the specific context of the questions asked.

A. Evaluation of User Experience Using System Usability Scale

User experience is crucial, which is why the System Usability Scale (SUS) was employed. It is a tool that is widely used to measure the usability of various systems and products. This scale provides a quick and reliable usability assessment from the user's perspective [24]. For this project, we used the SUS tool to assess the usability of the system we implemented. This evaluation involved ten individuals from the Virtual Reality department at the Centro de Investigación, Innovación y Transferencia Tecnológica (CIITT) with substantial expertise in immersive technologies. Their feedback provided a comprehensive evaluation of the system's usability. The results of this evaluation are shown in Fig. 7.

The system achieved an impressive average score of 91.00 on the SUS scale for usability and user satisfaction. The high SUS scores obtained from participants indicate exceptional levels of user satisfaction, rapid adoption, and ease of use, equivalent to A+ grades. This positive feedback suggests the significant potential of the adoption and application of this system in academic and research environments.

The integration of AI-powered assistants in academic environments presents numerous ethical considerations that necessitate thorough examination. While this technology



Personalized Assistant

Write a question:

Tell me the price for each semester of "software engineering"

Assistant: The costs for each semester of the "Software Engineering" program are as follows:

- 1st Cycle: $1,402.49
- 2nd Cycle: $1,402.49
- 3rd Cycle: $1,495.99
- 4th Cycle: $1,495.99
- 5th Cycle: $1,589.49
- 6th Cycle: $1,589.49
- 7th Cycle: $1,776.49
- 8th Cycle: $1,776.49

Additionally, there is a matriculation fee of $187 for the "Software Engineering" program 【24:0†source】.

User: Tell me the price for each semester of "software engineering"

Personalized Assistant

Write a question:

I would like to postulate for a scholarship, what are the requirements?

Assistant: To apply for a scholarship, you would need to go through the pre-registration and registration processes. Here are the general steps based on the information available:

1. **Pre-registration:**
   - The aspirant or applicant wishing to apply for the leveling course offered by the University must have a high school diploma or equivalent as a prerequisite 【31†source】.
2. **Registration:**
   - Upon completing the pre-registration process, the aspirant must proceed to pay the fees set by the institution to be considered officially enrolled in the leveling course 【31†source】.
3. **Admission Process:**
   - The admission process consists of leveling and induction for careers with high and low demand, respectively. The duration of these processes is 100 hours.
   - In the case of high-demand careers, aspirants will undergo a leveling course

Fig. 6. Assistant history chat about academic information related to professional careers availability, tuition costs, payment options, and scholarship requirements. The assistant and user inputs are labeled on different fields.

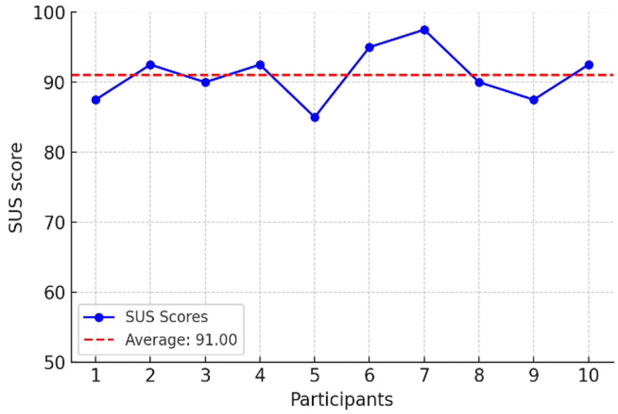


Fig. 7. Statistical illustration displaying SUS scores from ten participants. The X-axis represents the participants numbered from one to 10, while the Y-axis shows the scores ranging from 50 to 100.

has the capacity to transform organizations and the availability of academic data, it also introduces substantial issues regarding bias, privacy, and data accuracy.

B. Potential Biases

One of the significant challenges associated with using AI, particularly LLMs, is the potential existence of inherent biases. Such biases can result from the training data used in developing the models. If the training data contains biases, they could manifest in the responses that are generated by the assistant trained on that data. For instance, an LLM that is primarily trained on data in various languages may not offer accurate responses for users. To address these biases, it is crucial to utilize representative datasets during model training and to implement adjustment techniques to the document content to enhance the assistant's comprehension.

C. Data Privacy

Privacy is a major concern when using AI assistants in academic settings. These assistants have access to sensitive student and staff information, and it is crucial to ensure that this data is handled securely and confidentially. To mitigate these concerns, the assistant's responses are restricted to information that is found exclusively within academic documents, which must be exhaustively reviewed before being loaded to the assistant using sophisticated prompt engineering techniques.

D. Disinformation and Information Accuracy

The accuracy of the information provided by AI assistants is a crucial aspect of their effectiveness and reliability. LLMs, while powerful, are not infallible, and they can generate incorrect or misleading answers. This situation can be particularly problematic in academic contexts where it is vital to have information accuracy in all cases. It is therefore paramount to implement information verification and validation mechanisms. Assistants based on OpenAI are capable of citing verifiable sources. Supervised learning techniques and continuous feedback can also be used to improve the accuracy and reliability of models.

The results of the comprehensive assessments herein underscore the efficacy and potential impact of AI-powered assistants in facilitating efficient access to academic information. Continuous refinement and innovation are vital to maximizing utility and addressing the evolving user needs in academic environments.

E. Code Availability

The code generated for this work can be found in https://github.com/xr-lab-ucacue/Academic-Assistant.git.

V. CONCLUSION AND FUTURE WORK

Advancements in new advanced AI-powered assistants present a potential shift in the landscape of academic support services, as they can potentially replacing certain human tasks. This transition also opens up new opportunities for employment and innovation, emphasizing the dynamic nature of technological advancement in academic environments.

After performing this research focusing on an AI-powered assistant for academic information using a knowledge retrieval tool from OpenAI, it is clear that integrating NLP technologies can greatly enhance the accessibility and management of academic documents. The results show that this technology has the potential to revolutionize the way we interact with academic information.

Future research should strive to explore and compare multiple LLMs that are publicly available and restricted to evaluate the scalability of the proposed methodology.

Prominent models include OpenAI's GPT-4, open LLMs, and other closed LLMs such as Claude. Such a comparison would demonstrate the robustness of our methodology. It would also help identify the most suitable models for various academic tasks based on their performance and cost. The comparative approach will focus on implementing the same prompting across different LLMs, which will allow for a direct assessment of each model's ability to handle specific academic tasks. This method will facilitate the identification of the relative strengths and weaknesses of each LLM, ultimately providing a solid basis for selecting the most efficient and effective model for academic applications.

The comparative analysis will also include a thorough evaluation of the performance factors, such as response accuracy, consistency in content generation, and the ability to handle complex and contextually rich queries. The costs of using each LLM, in terms of the computational resources required as well as licensing and model access, will also be considered.

Despite the implications of this work for designing better LLM-powered assistants, this assistant has some limitations that can guide future research directions. First, due to the assistant API latency issues, because it is still in beta version, the integration of LLMs into a personalized assistant resulted in errors that would ideally be avoidable. Future works should explore alternative LLMs and their performance in solving user queries.

CONFLICT OF INTEREST

The authors have declared that no competing interests exist.

ACKNOWLEDGEMENTS

The XR LAB Department in the Centro de Investigación, Innovación y Transferencia Tecnológica (CIITT) fully supports this work, providing the necessary equipment to develop it.

REFERENCES

1. M. R. King, "The future of AI in medicine: a perspective from a Chatbot," *Annals of Biomedical Engineering*, vol. 51, pp. 291-295, 2023. https://doi.org/10.1007/s10439-022-03121-w
2. A. Pham Duy and H. Le Thanh, "A question-answering system for vietnamese public administrative services," in *Proceedings of the 12th International Symposium on Information and Communication Technology*, Ho Chi Minh, Vietnam, 2023, pp. 85-92. https://doi.org/10.1145/3628797.3628965
3. D. Mhlanga, "Open AI in education, the responsible and ethical use of ChatGPT towards lifelong learning," in *FinTech and Artificial Intelligence for Sustainable Development.* Cham, Switzerland: Springer, 2023, pp. 387-409. https://doi.org/10.1007/978-3-031-37776-1_17
4. H. Abu-Rasheed, M. H. Abdulsalam, C. Weber, and M. Fathi, "Supporting student decisions on learning recommendations:

an LLM-based Chatbot with knowledge graph contextualization for conversational explainability and mentoring," 2024 [Online]. Available: https://arxiv.org/abs/2401.08517.

5. J. Prather, P. Denny, J. Leinonen, B. A. Becker, I. Albluwi, M. E. Caspersen, et al., "Transformed by transformers: Navigating the AI coding revolution for computing education: an iticse working group conducted by humans," in *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education*, Turku, Finland, 2023, pp. 561-562. https://doi.org/10.1145/3587103.3594206
6. M. Perkins, "Academic Integrity considerations of AI large language models in the post-pandemic era: ChatGPT and beyond," *Journal of University Teaching & Learning Practice*, vol. 20, no. 2, article no. 7, 2023. https://doi.org/10.53761/1.20.02.07
7. OpenAI, "Introducing ChatGPT," 2022 [Online]. Available: https://openai.com/index/chatgpt/.
8. R. Tang, Y. N. Chuang, and X. Hu, "The science of detecting LLM-generated texts," 2023 [Online]. Available: https://arxiv.org/abs/2303.07205.
9. V. D. Kirova, C. S. Ku, J. R. Laracy, and T. J. Marlowe, "Software engineering education must adapt and evolve for an LLM environment," in *Proceedings of the 55th ACM Technical Symposium on Computer Science Education*, Portland, OR, USA, 2024, pp. 666-672. https://doi.org/10.1145/3626252.3630927
10. S. Quevedo, F. Merchan, R. Rivadeneira, and F. X. Dominguez, "Evaluating apache openwhisk-faas," in *Proceedings of 2019 IEEE 4th Ecuador Technical Chapters Meeting (ETCM)*, Guayaquil, Ecuador, 2019, pp. 1-5. https://doi.org/10.1109/ETCM48019.2019.9014867
11. S. Quevedo, F. Domingez, and E. Pelaez, "Detecting multi thoracic diseases in chest X-Ray images using deep learning techniques," in *Proceedings of 2023 IEEE 13th International Conference on Pattern Recognition Systems (ICPRS)*, Guayaquil, Ecuador, 2023, pp. 1-7. https://doi.org/10.1109/ICPRS58416.2023.10179041
12. S. Pati, U. Baid, B. Edwards, M. Sheller, S. H. Wang, G. A. Reina, et al., "Federated learning enables big data for rare cancer boundary detection," *Nature Communications*, vol. 13, article no. 7346, 2022. https://doi.org/10.1038/s41467-022-33407-5
13. L. J. Jacobsen and K. E. Weber, "The promises and pitfalls of ChatGPT as a feedback provider in higher education: an exploratory study of prompt engineering and the quality of AI-driven feedback," 2023 [Online]. Available: https://doi.org/10.31219/osf.io/cr257.
14. J. D. Velasquez-Henao, C. J. Franco-Cardona, and L. Cadavid-Higuita, "Prompt engineering: a methodology for optimizing interactions with AI-language models in the field of engineering," *Dyna*, vol. 90, no. 230, pp. 9-17, 2023. https://doi.org/10.15446/dyna.v90n230.111700
15. S. Ekin, "Prompt engineering for ChatGPT: a quick guide to techniques, tips, and best practices," 2023 [Online]. Available: https://doi.org/10.36227/techrxiv.22683919.v2.
16. B. Chen, Z. Zhang, N. Langrene, and S. Zhu, "Unleashing the potential of prompt engineering in large language models: a comprehensive review," 2023 [Online]. Available: https://arxiv.org/abs/2310.14735.
17. OpenAI Platform, "OpenAI documentation," c2024 [Online]. Available: https://platform.openai.com/docs/overview.
18. OpenAI, "Pricing," 2024 [Online]. Available: https://openai.com/api/pricing/.
19. Mistral AI, "Pricing and rate limits," c2024 [Online]. Available: https://mistral.ai/technology/#pricing.
20. Google Cloud, "Precios de Vertex AI," c2024 [Online]. Available: https://cloud.google.com/vertex-ai/generative-ai/pricing?hl=es.
21. Anthropic PBC, "Claude API," c2024 [Online]. Available: https://www.anthropic.com/api.
22. OpenAI, "Assistant API (v2) FAQ," c2024 [Online]. Available: https://help.openai.com/en/articles/8550641-assistants-api.
23. Snowflake Inc., "Streamlit documentation," c2024 [Online]. Available: https://docs.streamlit.io/.
24. J. Brooke, "SUS: a quick and dirty usability scale," in *Usability Evaluation in Industry*. London, UK: CRC Press, 1996, pp. 189-194. https://doi.org/10.1201/9781498710411

Pedro Fernando Álvarez https://orcid.org/0009-0007-2025-1068

Pedro Fernando Álvarez is a student at the Faculty of Informatics, Computer Science, and Technological Innovation (ICCIT) at Universidad Católica de Cuenca. Throughout his academic training, he has demonstrated a strong commitment to research and innovation in artificial intelligence and innovative software development.

Sebastián Quevedo https://orcid.org/0000-0001-5585-0270

Sebastián Quevedo is a Ph.D. candidate in Computer Science at the Escuela Superior Politécnica del Litoral, Ecuador. Since 2012, he has been a tenured professor in the Software Engineering program at the Universidad Católica de Cuenca. Additionally, he has been a guest lecturer in the Master's program in Cybersecurity since 2022. His research interests include deep learning applied to computer vision tasks and natural language processing.