Retrieval Augmented Generation in Large Language Models: Development of AI Chatbot for Student Support

Dijana Oreški *,[1] Dino Vlahek [2,†]

[1] *University of Zagreb, Faculty of Organization and Informatics, Pavlinska 2, Varaždin, 42000, Croatia*
[2] *University of Maribor, Faculty of Electrical Engineering and Computer Science, Koroška cesta 46, SI-2000 Maribor, Slovenia*

Abstract

Large Language Models (LLMs) are being employed in various domains to support different tasks. There are many challenges when working with LLMs, such as hallucination and domain knowledge gaps. Retrieval Augmented Generation (RAG) has emerged as one of the best paradigms for enabling LLMs to access domain-specific data and as a mechanism for mitigating hallucinations. In this paper, we are employing RAG and investigating how large language models use repositories of pre-existing knowledge to enhance the quality and relevance of generated responses. The focus of the paper is the development of real-world RAG applications in the educational domain. The research aims to develop an AI-based chatbot for university students that could answer students` frequently asked questions. Such a chatbot solves several challenges faced by the faculty administration and guides the improvement of the student's study experience. In the implementation, a low-code approach of Flowise AI is used. As a result, the prototype is developed. From the domain point of view, the prototype represents a step toward an AI-powered educational system that has the potential to further enhance the level of individualized educational support. From the technological point of view, such implementation emphasizes the benefits of LLM and RAG integration and the advantages of generative artificial intelligence.

Keywords

Generative artificial intelligence, large language models, retrieval augmented generation, AI chatbot, student support.

1. Introduction

Generative artificial intelligence and LLM are at the forefront of innovation today. These systems demonstrate great abilities in creating content and natural language processing (NLP), revolutionizing various fields of application. LLM has various advantages: (i) solving most NLP tasks using just instructions and a few examples, (ii) performing math and logical reasoning, (iii) achieving human-level on standardized tests (SAT, LSAT, GRE) [1]. Considering this, there are various LLM applications today, such as: (i) contextual search, Q&A and chatbots, (ii) text summarization and translation, (iii) creative writing, (iv) reading, writing and fixing code. On the other hand, LLM has various deficiencies. They are prone to making up facts (hallucinations) and generating biased content serving as uninterpretable black boxes. There is a need to make LLM deployments trustworthy and reliable. One of the main questions arising here is how to reduce model hallucination. The answer is to incorporate external knowledge sources into LLM, which can be achieved by following one of the two approaches. The first one is fine-tuning of LLM which refers to the process of retraining LLM on smaller custom datasets. This approach requires high computing power and the need to fine-tune it again once new data comes in. The second approach is retrieval augmented generation (RAG). RAG is a process of retrieving information from external knowledge sources with the help of LLM [2]. It refers to retrieving of the relevant information from external knowledge bases before answering questions with LLM. Combining RAG with LLM offers a powerful approach to enhance various fields of application. Education is in the front. This research is focused on the development of an artificial intelligence chatbot for student support using the combination of

Proceedings for the 15th International Conference on e-Learning 2024, September 26-27, 2024, Belgrade, Serbia

* Corresponding author.
† These authors contributed equally

dijana.oreski@foi.h (D. Oršeki); dino.vlahek1@um.si (D. Vlahek);
0000-0002-3820-0126 (D. Oršeki); 0000-0002-3911-8685 D. Vlahek);

LLM and RAG. The aim of this research is twofold. The aim of this research is twofold: firstly, to introduce artificial intelligence-based chatbots at the university level, and secondly, to combine LLM with RAG using a low-code approach.

Reminder of the paper is as follows. The second section explores RAG concepts and types, focusing on the educational domain. The third section focuses on the agents' requirements, explaining functional and non-functional bot requirements along with a use case diagram. The fourth section explains chatbot implementation in the Flowise AI tool by explaining the main components and describing testing of the developed chatbot. The final section concludes the paper by indicating contributions and guidelines for future research.

2. Retrieval augmented generation in education

2.1. Retrieval augmented generation foundations

LLM can reason about diverse topics, but knowledge is limited to the public data available up until their training cutoff date. To create AI applications that can process private data or information introduced after the model's training period, augmenting the model's knowledge with relevant information is necessary. This process of incorporating appropriate information into the model prompt is known as RAG. RAG is a technique that enhances the knowledge of LLM by incorporating additional data. RAG represents a solution to limited context windows. There are three phases in the RAG process. First is the retrieval phase, where the model searches through databases or document collections to identify the most relevant facts and passages for the given prompt or user question. Second is the augmented phase, where the retrieved snippets of external knowledge are appended to the original user input, augmenting the context. The last, i.e., a generation phase, is where the language model analyzes expanded prompts to produce a response. It references both the retrieved information and its internally trained patterns to formulate an informative and natural answer.
LLM is trained on massive text datasets, enabling them to generate human-quality text, translate languages, write different kinds of creative content, and answer questions in an informative way. However, they have limitations in accessing and grounding their responses in specific information. RAG addresses this limitation by connecting LLMs to external knowledge bases. When the user ask a question, RAG retrieves relevant information from these sources and provides it as context to the LLM, enabling more accurate and grounded responses. Key Components of the RAG are:

- Language model,
- Vector store (database),
- Retriever,
- Embedder, and
- Indexer/loader.

Language model represents a foundation of RAG architecture, a pre-trained language model responsible for text generation. GPT-3, Llama 2, and Google BERT have language comprehension and synthesis capabilities, enabling them to engage in conversational dialogues. Vector store is central to the retrieval functionality [3]. It is a vector store database that stores document embeddings for efficient similarity searches. This allows for rapid identification of relevant contextual information. The Retriever module utilizes the vector store to locate documents and passages that augment the prompts [4]. Neural retrieval approaches excel at semantic matching. Embedder encodes source documents into vector representations that the retriever can consume and populate the vector store [5]. An indexer is needed since robust pipelines ingest and preprocess source documents, breaking them into manageable passages for embedding and efficient lookup [6]. By using these core components, RAG systems empower language models to access vast knowledge resources, enabling generation and improved responses. There are various RAG categorizations; based on retrieval methods, integration with LLM, knowledge sources, or applications. Hereinafter, we apply iterative RAG regarding LLM integration in which retrieval and generation happen in multiple steps [7]. The LLM's initial output can be used to refine the retrieval query, leading to more relevant context for subsequent generation [7].

Based on the knowledge source, our RAG LLM implementation is a closed domain, focusing on specific domains or tasks and retrieving information from specialized datasets like scientific papers, legal documents, or medical records. Our application is a dialogue system where RAG enables chatbots and conversational agents to access external knowledge and provide more informative and contextually relevant responses [8].

2.2. Educational needs for retrieval augmented generation

This paper focuses on developing AI chatbots within the educational sector, incorporating key processes within the faculty. This domain includes rules, policies, procedures, and general information relevant to students and their academic experience. Those are:

(i) administrative rules: information regarding enrollment, withdrawal, transfer from other faculties, etc;
(ii) academic rules: questions related to class schedules, exams, grading, plagiarism, etc.;
(iii) documents and applications: tuition fees, scholarships, payment methods, student status verification documentation, etc;
(iv) student rights and obligations: student rights and responsibilities, disciplinary procedures, appeals procedures;
(v) other information: questions related to student housing, food services, healthcare, library, sports, and cultural activities.

The purpose of developing an AI-based chatbot within higher education is to provide students with quick, accurate, and easily accessible information about faculty rules and policies. The chatbot should reduce the burden on administrative staff, increase communication efficiency, and enhance the overall student experience. Objectives of such AI-based chatbot are:

(i) improving access to information: providing students with 24/7 access to information through a simple interface and reducing the time needed to get answers to frequently asked questions.
(ii) automating administrative processes: automating responses to common questions to reduce the burden on administrative staff and increasing the efficiency of information management within the faculty.
(iii) personalizing customer support based on the specific needs and status of the student by ensuring relevant information in the context of individual inquiries and increasing student satisfaction.

3. Research methodology

For chatbot implementation, we have used FlowiseAI, an open-source visual tool used to build customized LLM flows [9]. This is the leading no-code platform for LangChain-based LLM workflow development which aims to make AI accessible to users with minimal technical expertise. FlowiseAI enables faster development cycles and reduces the need for extensive coding skills.

3.1. Functional and non-functional requirements

In the first phase of chatbot development, we have specified chatbot requirements, both functional and non-functional. Table 1 presents nine functional requirements along with their description.

Table 1.
Functional requirements

Requirement	Description
Initiating interaction with the user.	The chatbot must be capable of automatically initiating interaction with the user as soon as they access the page. The chatbot should offer an introductory greeting and introduce itself to the user.
Built-in ticketing system.	When the chatbot cannot answer the inquiry, it should return the contact details of the responsible person to the user.
Options for inquiries at the beginning of the conversation.	The chatbot should offer several predefined options for inquiries (e.g., exam rules, graduation rules, financial aid, etc.).
Polite handling of user inquiries.	If the chatbot doesn't know the answer to the user's query, a ticket should be automatically created, and the user should be notified.
Multilingual support.	The chatbot must support multiple languages (English and Croatian) to communicate with users who speak different languages (exchange students).
Answering open-ended questions based on provided documents.	The chatbot should be able to search and respond to questions based on information from provided documents.
Taking into acccount the conversation context.	The chatbot should be capable of storing the context of the interaction so that the conversation can continue without repeating information.
Answering open-ended questions based on the website.	The chatbot should be able to answer more general questions about the faculty based on information from the website.
Receiving feedback on individual messages.	The chatbot should give the user the option to rate the quality (relevance) of the message.

Table 2 describes five non-functional requirements with their descriptions.

Table 2.
Non-functional requirements

Requirement	Description
24/7 availability.	The chatbot must be available to users 24 hours a day, 7 days a week.
Acceptable response speed.	The chatbot should respond to user queries within seconds to ensure a fast and efficient user experience.
Unambiguous responses.	The chatbot's responses must be clear and precise so that users can easily understand the information.
Ease of use.	The chatbot should have an intuitive and simple user interface that allows easy use without the need for additional instructions.
Integration with the faculty website.	The chatbot must be integrated with the official faculty website so that users can easily access the chatbot and relevant information.

These requirements provide the foundation for developing an effective chatbot that can help students navigate faculty rules. The focus is on functionality that enables students to quickly and easily access the necessary information while reducing human resource involvement only when necessary. Non-functional requirements ensure that the chatbot is available, fast, and easy to use.

3.2. Use case diagrams

Use case diagrams show the behavior of the system from the user's perspective. The use case diagram of our chatbot is presented in Figure 1.

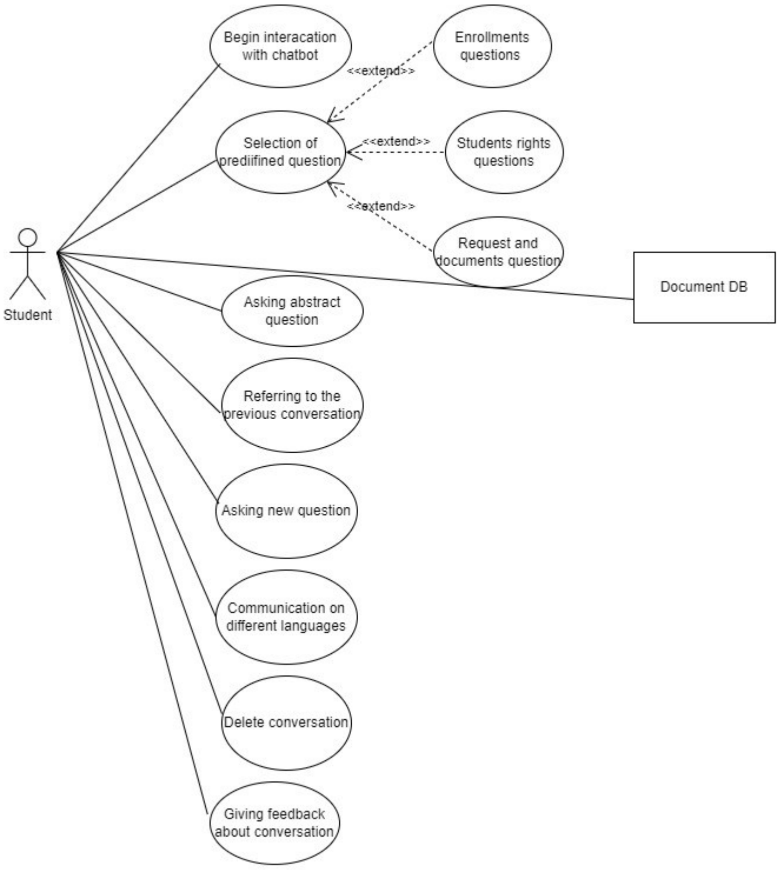

Begin interacation with chatbot
Enrollments questions
<<extend>>
Selection of prediifined question
<<extend>>
Students rights questions
<<extend>>
Request and documents question
Student
Document DB
Asking abstract question
Referring to the previous conversation
Asking new question
Communication on different languages
Delete conversation
Giving feedback about conversation

Figure 1: Use case diagram

The actor is a student. There are eight main use cases. The main idea behind each of the use cases is as follows.

1. **User communicates in different languages:**
   - **Description:** At the start of the interaction, the chatbot communicates in English and adapts its language to the user during the interaction.
   - **Example:**
     User accesses the faculty website
     User asks a question in Croatian
     „Kakav je postupak prelaska s drugog fakulteta?“
     Chatbot continues communication in Croatian
     “Prijelaz na studij informatike iste razine moguć je s nekog drugog srodnog studija unutar Fakulteta, s nekog drugog srodnog studija unutar Sveučilišta u Zagrebu, ili s nekog drugog sveučilišta u skladu s općim aktom Fakulteta. O uvjetima prijelaza odluku donosi Fakultetsko vijeće. Iznimno se može odobriti prijelaz studentima koji ne zadovoljavaju opće uvjete ukoliko je prijelaz potreban zbog teže bolesti, preseljenja obitelji, obveze treninga vrhunskih sportaša ili nekog drugog opravdanog razloga”
2. **User begins interaction with the chatbot:**
   - **Description:** The user initiates interaction with the chatbot through the faculty website. The chatbot automatically starts the conversation by providing the user

with several predefined options for inquiries. The user then selects one of the offered questions to get the desired information.

- **Example:**
  Chatbot automatically sends an initial message and predefined options:
  "Hello, I am your virtual assistant. How can I help you today?"
  User selects one of the offered options or asks an open-ended question.

3. **Selecting one of the predefined questions:**

   - **Description:** The user initiates interaction with the chatbot through the faculty website. The chatbot automatically starts the conversation by providing the user with several predefined options for inquiries. The user then selects one of the offered questions to get the desired information.

   - **Example:**
     Chatbot automatically sends an initial message and predefined options
     "Good day! I am your virtual assistant. How can I help you today?

     1. What are the requirements for graduating from college?

     2. What are the requirements for enrollment in the following year?

     3. What is the procedure for transferring from another college?"
        User selects the first option
        Chatbot responds to the predefined question
        "Conditions for the completion of studies include the creation of a thesis and its defense before the commission for the thesis in accordance with the Regulations on the thesis. Also, during the course of the studies, the student should obtain a sufficient number of ECTS points needed to complete the studies. A note is that the student does not need to pass all the enrolled electives if he achieves the required number of ECTS credits."

4. **User asks an open-ended unstructured question:**

   - **Description:** The user asks an open-ended unstructured question instead of selecting a predefined question, and the chatbot uses its knowledge base, given documents, and faculty website to find a relevant answer and provide the user with the requested information.

   - **Example:**
     User accesses the faculty website.
     Chatbot automatically sends an initial message
     User decides to enter their own question:
     "What are the procedures for transferring from another faculty?"
     Chatbot searches its knowledge base, provided documents, and faculty website, and responds:
     "Transferring from another faculty requires you to submit a request to the student office. You need to provide an official transcript of grades and fill out a credit transfer form. The approval process may take some time, so it is advisable to submit the required documentation as early as possible. If you need additional information or assistance, feel free to ask!"

5. **User asks an abstract question:**

   - **Description: The** user asks a general and abstract question about the faculty or the university that is not directly related to specific rules or documents, and the chatbot searches the faculty website to find a relevant answer and provide the user with the requested information.

- **Example:**
User accesses the faculty website.
Chatbot automatically sends an initial message
User decides to enter their own question
"Where is the Faculty of Organization and Informatics located?"
Chatbot searches the faculty website for the answer and responds:
"The Faculty of Organization and Informatics is located in Varaždin, Pavlinska 2."

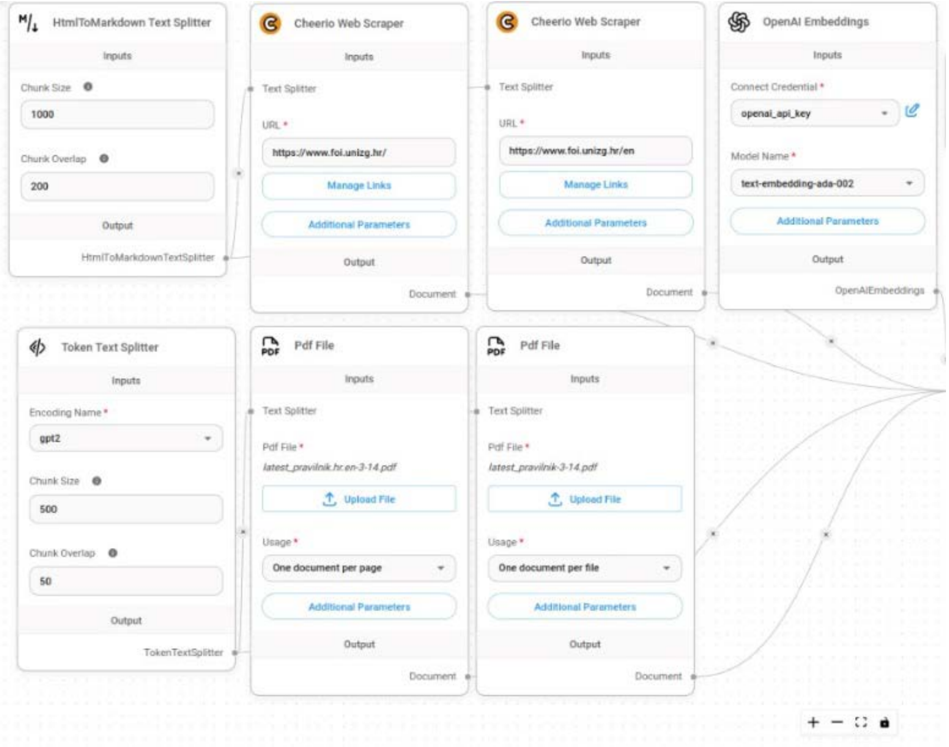
6. **Feedback collection:**

- **Description**: The user evaluates the quality of the chatbot's answer and the chatbot stores the feedback and automatically adapts and improves its knowledge base.

- **Example**:
User accesses the faculty website.
Chatbot automatically sends an initial message and predefined options.
User selects one of the options or asks an open-ended question.
Chatbot answers the user's question.
Chatbot then asks the user to evaluate the quality of the response:
"Was this information helpful? Please rate your experience."
User provides a rating or feedback
Chatbot adapts and improves its knowledge base based on the feedback.

4. Development, testing, and evaluation of AI-based chatbot

The RAG component implemented in the AI chatbot is in the chatbot's knowledge base, consisting of a website and the faculty's regulations in PDF format. Bilingual documents and two versions of the FOI website (Croatian and English versions) have been loaded to implement RAG. Data from the websites was collected using the Cheerio Web Scraper node, after which the HtmlToMarkdown Text Splitter was used to convert HTML to Markdown format and then split it into individual documents according to titles in MD format. For PDF documents, we used the PDF file node for loading and the Token Text Splitter for document separation, which divides the raw text string into smaller parts based on BPE tokens. After tokenization, the created tokens are separated into fragments of a specific size. Finally, the BPE tokens within each unit are rejoined to create the actual text. The text split in this way is stored in the Pinecone vector database in the form of 1536-dimensional vectors created using the OpenAI Embeddings node. We then used ChatOpenAI as input to the Conversational Retrieval QA Chain, which also receives vectors of units from the Pinecone database that are relevant to the user's query. Memory has also been implemented in the form of a Conversational Summary Memory node and cache memory for the language model.



HtmlToMarkdown Text Splitter: Chunk Size 1000; Chunk Overlap 200

Cheerio Web Scraper: URL * https://www.foi.unizg.hr/

Cheerio Web Scraper: URL * https://www.foi.unizg.hr/en

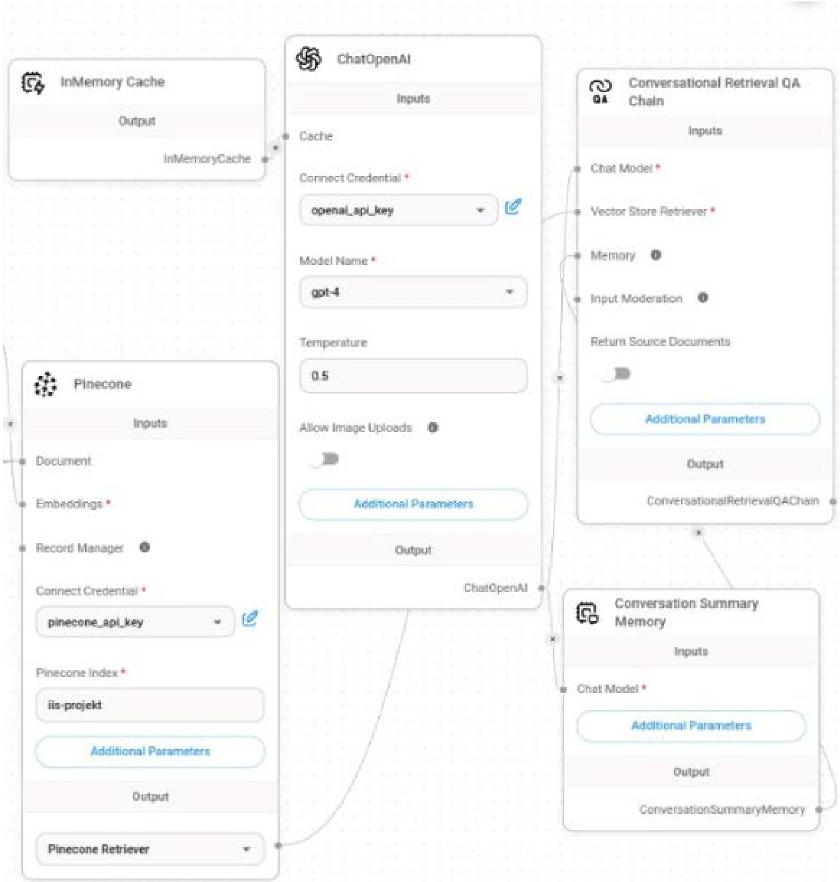
OpenAI Embeddings: Connect Credential * openai_api_key; Model Name * text-embedding-ada-002

Token Text Splitter: Encoding Name * gpt2; Chunk Size 500; Chunk Overlap 50

Pdf File: Pdf File * *latest_pravilnik.hr.en-3-14.pdf*; Usage * One document per page

Pdf File: Pdf File * *latest_pravilnik-3-14.pdf*; Usage * One document per file

Figure 2a: Chatbot workflow – part 1



ChatOpenAI: Connect Credential * openai_api_key; Model Name * gpt-4; Temperature 0.5

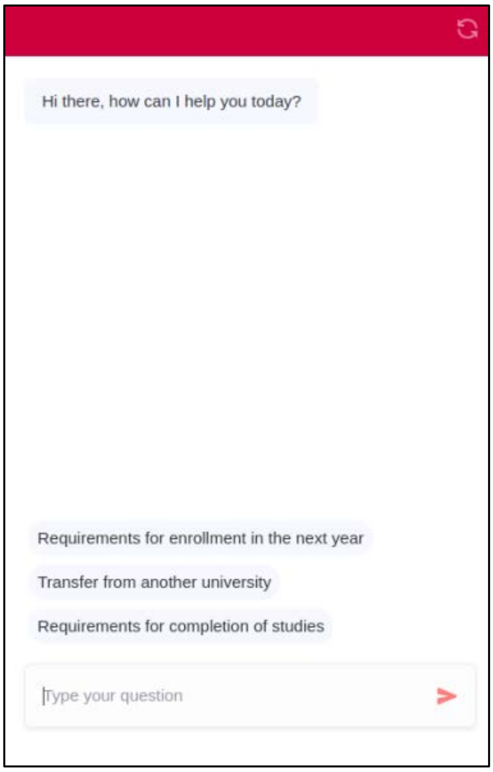
Pinecone: Connect Credential * pinecone_api_key; Pinecone Index * iis-projekt; Output Pinecone Retriever

Figure 2b: Chatbot workflow – part 2

The whole process is presented in Fig.2a and 2b. In the final phase, chatbot testing was performed. Testing was performed with one group of 15 students.

When initially opening the chatbot, the user is greeted with the chatbot's initial welcome message: "Hi there, how can I help you today?". Additionally, three predefined questions are displayed, which the user can select for an answer (Figure 3*a*).

In the first case, the user selects one of the predefined questions, to which the chatbot responds. (Figure 3*b*). The chatbot is capable of communicating in two languages: English and Croatian, and will adapt to the user's input. In the image below, we can also observe the chatbot's ability to answer open-ended questions using appropriate documentation and links to the FOI website (Figure 3*c*).

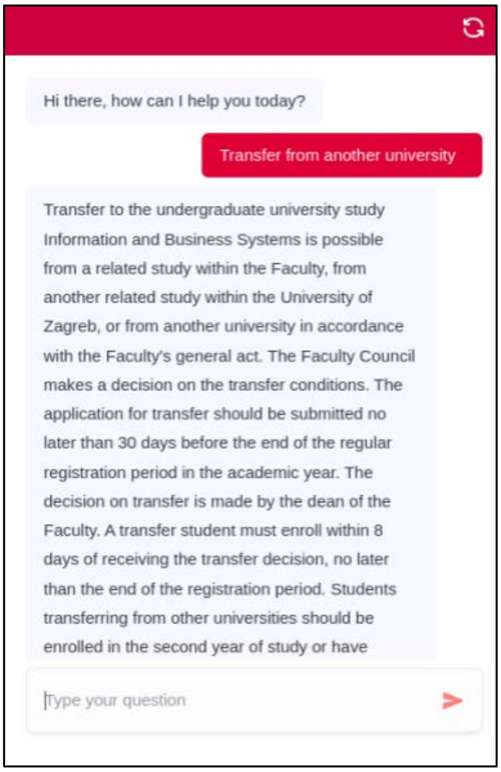


Hi there, how can I help you today?

Requirements for enrollment in the next year

Transfer from another university

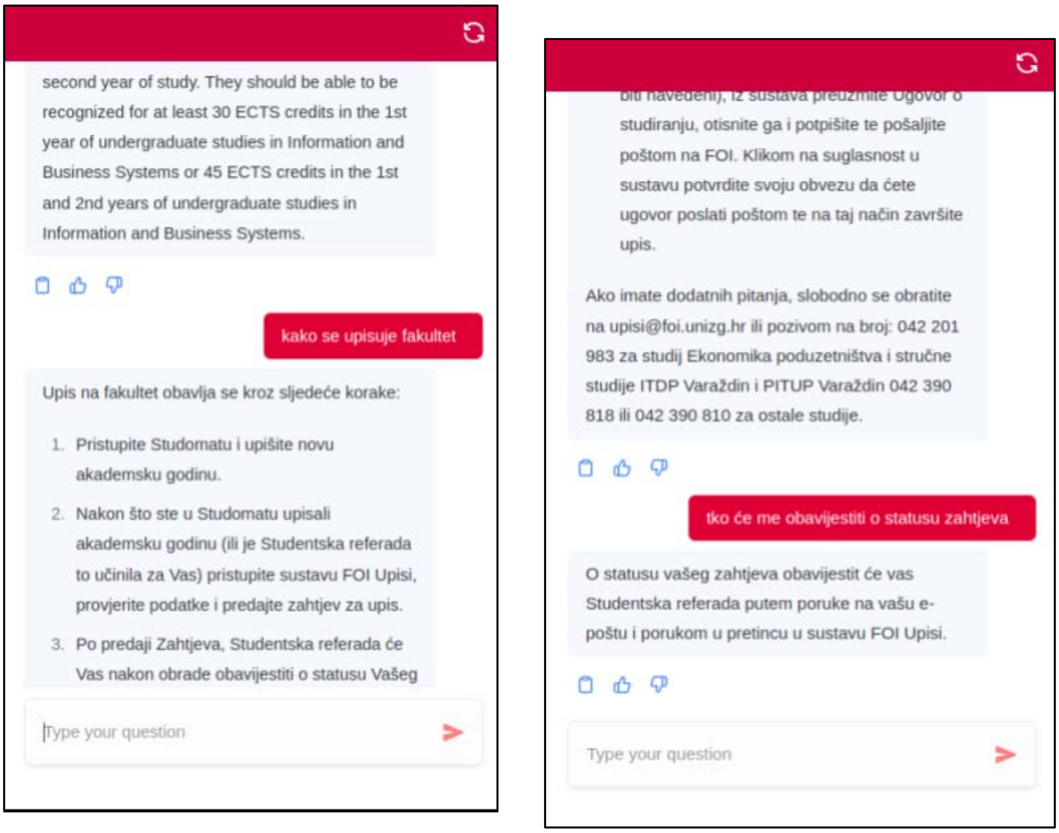
Requirements for completion of studies



Hi there, how can I help you today?

Transfer from another university

Transfer to the undergraduate university study Information and Business Systems is possible from a related study within the Faculty, from another related study within the University of Zagreb, or from another university in accordance with the Faculty's general act. The Faculty Council makes a decision on the transfer conditions. The application for transfer should be submitted no later than 30 days before the end of the regular registration period in the academic year. The decision on transfer is made by the dean of the Faculty. A transfer student must enroll within 8 days of receiving the transfer decision, no later than the end of the registration period. Students transferring from other universities should be enrolled in the second year of study or have



second year of study. They should be able to be recognized for at least 30 ECTS credits in the 1st year of undergraduate studies in Information and Business Systems or 45 ECTS credits in the 1st and 2nd years of undergraduate studies in Information and Business Systems.

kako se upisuje fakultet

Upis na fakultet obavlja se kroz sljedeće korake:

1. Pristupite Studomatu i upišite novu akademsku godinu.
2. Nakon što ste u Studomatu upisali akademsku godinu (ili je Studentska referada to učinila za Vas) pristupite sustavu FOI Upisi, provjerite podatke i predajte zahtjev za upis.
3. Po predaji Zahtjeva, Studentska referada će Vas nakon obrade obavijestiti o statusu Vašeg

biti navedeni), iz sustava preuzmite Ugovor o studiranju, otisnite ga i potpišite te pošaljite poštom na FOI. Klikom na suglasnost u sustavu potvrdite svoju obvezu da ćete ugovor poslati poštom te na taj način završite upis.

Ako imate dodatnih pitanja, slobodno se obratite na upisi@foi.unizg.hr ili pozivom na broj: 042 201 983 za studij Ekonomika poduzetništva i stručne studije ITDP Varaždin i PITUP Varaždin 042 390 818 ili 042 390 810 za ostale studije.

tko će me obavijestiti o statusu zahtjeva

O statusu vašeg zahtjeva obavijestit će vas Studentska referada putem poruke na vašu e-poštu i porukom u pretincu u sustavu FOI Upisi.

Figure 3: Chatbot testing

Chatbot has the ability to store the context when conversing with the user. If the user references a previous message in the conversation, it will recognize that reference and provide an appropriate response (Figure 3d).

5. Conclusion

This paper presented LLM and RAG integration in an educational setting. RAG, in conjunction with LLM, holds immense potential for transforming education. AI-based chatbot for higher education purposes presented here is only one example of how this approach can create effective learning experiences. However, it's crucial to address several challenges and ethical considerations to ensure responsible implementation. The first one refers to data bias. The quality of RAG-augmented LLMs depends heavily on the data they are trained on. Biases present in the training data can perpetuate and even amplify existing inequalities in education. The second one refers to the ethical implications. As with any AI application in education, careful consideration must be given to ethical implications, ensuring transparency, fairness, and student privacy. Deeper testing should be performed in order to evaluate quality of the chatbot.

Acknowledgment

This paper is supported by Croatian science foundation under the project SIMON: Intelligent system for automatic selection of machine learning algorithms in social sciences, UIP-2020-02-6312.

Declaration on Generative AI

During the preparation of this work, the author(s) used X-GPT-4 and Gramby in order to: Grammar and spelling check. Further, the author(s) used X-AI-IMG for figures 3 and 4 in order to: Generate images. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] Naveed, H., Khan, A. U., Qiu, S., Saqib, M., Anwar, S., Usman, M., ... & Mian, A. (2023). A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*.

[2] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., ... & Wang, H. (2023). Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

[3] Wang, X., Wang, Z., Gao, X., Zhang, F., Wu, Y., Xu, Z., ... & Huang, X. (2024). Searching for Best Practices in Retrieval-Augmented Generation. *arXiv preprint arXiv:2407.01219*.

[4] Salemi, A., & Zamani, H. (2024, July). Evaluating Retrieval Quality in Retrieval-Augmented Generation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 2395-2400).

[5] Zhang, P., Liu, Z., Xiao, S., Dou, Z., & Nie, J. Y. (2024, August). A Multi-Task Embedder For Retrieval Augmented LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 3537-3553).

[6] Fleischer, D., Berchansky, M., Wasserblat, M., & Izsak, P. (2024). RAG Foundry: A Framework for Enhancing LLMs for Retrieval Augmented Generation. *arXiv preprint arXiv:2408.02545*.

[7] Shao, Z., Gong, Y., Shen, Y., Huang, M., Duan, N., & Chen, W. (2023). Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. *arXiv preprint arXiv:2305.15294*.

[8] Li, H., Su, Y., Cai, D., Wang, Y., & Liu, L. (2022). A survey on retrieval-augmented text generation. *arXiv preprint arXiv:2202.01110*.

[9] Flowise AI, https://flowiseai.com/ , last accessed: 15th August 2024.