International Journal of Human Computing Studies

e-ISSN: 2615-8159
p-ISSN: 2615-1898

Volume: 8 Issue: 1 | April 2026
https://journals.researchparks.org/index.php/IJHCS

Article

Implementation of an Academic Information Chatbot Based on Retrieval-Augmented Generation (RAG) Using LLaMA 3.1

Mochamad Alfan Rosid*[1], Muhammad Saddam Heykal Bustomy[2], Suprianto[3], Hamzah Setiawan[4]

1,2,3,4 Informatics Study Program, University of Muhammadiyah Sidoarjo, Indonesia
* Correspondence: alfanrosid@umsida.ac.id

Abstract: Providing fast and accurate academic information is a fundamental requirement for universities. However, users often experience difficulties in efficiently searching for specific information on study program websites. This study aims to develop a Retrieval-Augmented Generation (RAG)-based academic information chatbot that is directly integrated with the Informatics Study Program website at Muhammadiyah University of Sidoarjo. The system was built using the LangGraph architecture, ChromaDB vector database, and LLaMA 3.1 model through the Groq API. Testing was conducted using the RAGAS framework to assess the quality of answers and load testing for system performance. The results showed that the system achieved a Context Precision score of 0.86 and a Faithfulness score of 0.77, indicating high relevance and accuracy of answers. In addition, the implementation of the Groq API resulted in an average response latency of 2.11 seconds with a 94% success rate in load testing. These findings indicate that the RAG-based chatbot is effective in overcoming website limitations in delivering academic information at the Informatics Study Program at Muhammadiyah University of Sidoarjo.

Keywords: Academic Chatbot; Retrieval-Augmented Generation (RAG); LLaMA 3.1; Groq API; LangGraph.

Citation: Rosid M. A., Bustomy M. S. H., Suprianto., Setiawan H. Implementation of an Academic Information Chatbot Based on Retrieval-Augmented Generation (RAG) Using LLaMA 3.1. International Journal of Human Computing Studies 2026, 8(1), 26-36.

Received: 20th Jan 2026
Revised: 13th Feb 2026
Accepted: 07th Mar 2026
Published: 22nd Apr 2026

1. Introduction

Providing fast and accurate academic information is a fundamental requirement in education, including within the Informatics Study Program at Universitas Muhammadiyah Sidoarjo (UMSIDA) [1]. Along with the development of Artificial Intelligence (AI), users now enjoy various conveniences in searching for digital information sources and learning materials on the internet [2]. The UMSIDA Informatics Study Program has utilized digital platforms such as its official website, social media, and online documents to convey academic information to students [3]. However, in reality, users still frequently encounter difficulties in efficiently finding the required information, particularly regarding practicum schedules, which are most frequently searched. Information that is actually fully available on the official study program website is often asked directly to the administration or relevant lecturers. This condition occurs because students are reluctant to browse through numerous web pages or do not know where the relevant information is located.

This problem indicates that the main challenge of academic information services in the UMSIDA Informatics Study Program lies not only in data availability but also in the delivery mechanism and ease of access [4]. Static websites have limitations in supporting two-way interactions or contextual information retrieval [5]. Users must read and navigate

through many pages to obtain specific answers. Therefore, a system capable of presenting academic information interactively, responsively, and in accordance with user needs is required [6]. One relevant solution to this problem is the utilization of Artificial Intelligence (AI)-based chatbots [7].

A chatbot is an interactive system that allows users to communicate with the system via text messages [8]. In the context of academic services in the UMSIDA Informatics Study Program, chatbots can act as virtual assistants that help users obtain information efficiently [9]. Along with the development of Large Language Models (LLMs), chatbots' ability to understand questions and generate natural answers has increased significantly [10]. However, LLMs tend to face limitations in tasks requiring factual knowledge, as the models only rely on internal parameters from previous training [11].

Several previous studies have proposed the Retrieval-Augmented Generation (RAG) approach, a method that combines document retrieval capabilities with the generative capabilities of Large Language Models (LLMs) [12]. A study by Denina Milasanti [13] developed a RAG-based chatbot for scientific article retrieval on the GARUDA portal, while a study by Harifa Nur'aeni [14] implemented a RAG chatbot for information services in a digital library. Both studies used the ROUGE metric as a chatbot performance evaluation tool, which was rated as "good enough," although the numerical ROUGE scores obtained were relatively low. Additionally, a study by Hidayat et al. [15] developed a RAG-based academic chatbot integrated with the Telegram platform, showing an increase in answer accuracy, but still facing obstacles regarding response latency. Generally, these studies demonstrate the potential of RAG chatbots in supporting academic information services, but also indicate challenges in the evaluation, efficiency, and integration aspects of the service.

Based on these prior studies, several research gaps can be identified. First, the use of the ROUGE metric indicates a potential mismatch between the evaluation metric used and the characteristics of question-answering tasks, considering that ROUGE was originally designed for text summarization tasks [16]. Second, some studies still rely on third-party platforms or separate applications, meaning that user access to the chatbot is not yet directly integrated into the official UMSIDA Informatics Study Program website. Third, the aspects of response latency and system efficiency remain constraints, especially when using proprietary LLMs with relatively high costs and response times. Therefore, this study focuses on developing a RAG-based academic information chatbot that is directly integrated into the UMSIDA Informatics Study Program website, utilizing official data sources structured in JSON format, and using the LLaMA 3.1 model via the Groq API to achieve faster response times. This approach is expected to produce a chatbot system that is more responsive, relevant, and in line with academic information service needs.

2. Materials and Methods

The methodology of this research is designed to build an academic information chatbot system for the Informatics study program based on Retrieval-Augmented Generation (RAG). The research stages include literature study, data collection, data preprocessing, vector construction, chatbot workflow design, and evaluation of answer quality. The entire research workflow is systematically arranged and visualized in the flowchart in Figure 2.1.

Figure 2.1. Research Stages.

Conceptual Framework of Retrieval-Augmented Generation (RAG)

Prior to data collection, this study began with a foundational review of Retrieval-Augmented Generation (RAG) theory as the conceptual basis for developing the academic information chatbot. This review focused on understanding the working principles of RAG, which combines the document retrieval process and answer generation using a Large Language Model. This basic understanding is necessary to determine the system workflow, select core components, and formulate the RAG integration strategy within the chatbot to ensure it can generate relevant, accurate answers sourced from official data [12]. The results of this theoretical study serve as a reference for the subsequent data collection and system development phases.

Data Collection

Data was obtained from two main sources: the official website of the Informatics Study Program and PDF documents containing general information about the study program. Data extraction from the website was carried out using web scraping techniques via LangChain's built-in libraries, such as WebBaseLoader, to extract text from pages containing academic information [17]. Meanwhile, data from PDF documents was sourced from official files, such as academic handbooks.

Data Preprocessing

All obtained data, from both the website and PDF documents, was converted into a structured JSON data format. Each data entry was structured into page_content containing the main text, and metadata which includes source information, category, title, and URL. This process was conducted to standardize the data format from various sources by separating essential elements such as source type, document title, and content body. This approach facilitates data handling and enhances consistency in subsequent processing stages. An example of the preprocessing and conversion results within the JSON structure is shown in Table 2.1. Next, the JSON-formatted data was processed into text segments (chunks) with a maximum size of 4000 characters and an overlap of 200 characters. This chunking was performed to maintain contextual continuity between text segments and reduce the risk of losing critical information during vector embedding generation [18].

Table 2.1 Example of JSON Data Structure.

JSON Element	Content
page_content	Informatics Online Letter Administration Service. Students can apply for Active Student Letters, Internship Letters, Research, and Final Projects through this form.
metadata.source	https://informatika.umsida.ac.id/layanan-kemahasiswaan/
metadata.category	Service
metadata.title	Online Letter Form
metadata.direct_link	https://docs.google.com/forms/d/1ijKTVs1T546WU__zqqEc9JeSfj2HoGVWFqhcGaJr-bs/viewform?edit_requested=true

Vector Construction

After the preprocessing phase was completed, the text chunks were transformed into numerical representations (vector embeddings) that can be used for semantic searching [19]. This embedding process utilized the nomic-embed-text model [20] due to its ability to generate vectors that capture meaning and context precisely, and its compatibility with the developed system architecture. The process of converting text into vector representations can be seen in Table 2.2.

Table 2.2 Example of JSON Data Conversion to Vector Representation.

Component	Example
page_content (JSON)	Informatics Online Letter Administration Service. Students can apply for Active Student Letters, Internship Letters, Research, and Final Projects through this form.
Metadata	Service, Online Letter Form
Vector Embedding	$[0.021, -0.134, 0.587, \ldots, 0.042]$

The resulting embeddings were then stored and managed using ChromaDB as the vector database. ChromaDB was chosen due to its advantages in high-speed semantic searching, structured storage, and excellent integration with Retrieval-Augmented Generation (RAG) [21]. Through this vector-based storage, the system can efficiently perform semantic searches to find the text chunks most relevant to the user's query.

Chatbot System Development

The development process began with designing the chatbot workflow using LangGraph. LangGraph was selected because of its capability to manage process flows in the form of a Directed Acyclic Graph (DAG) [22], allowing each stage—from receiving a query, retrieving data, to generating an answer—to be managed modularly and structurally.

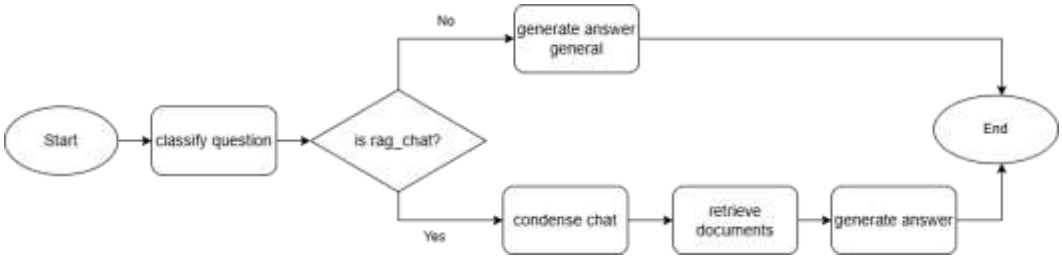

Figure 2.2. LangGraph Node Workflow.

Figure 2.2 illustrates the chatbot workflow using LangGraph. The user's question is classified (classify question) to determine whether it can be answered directly by the model (generate answer general) or requires document retrieval. If retrieval is required, the query is summarized (condense chat), then relevant documents are retrieved from the vector store (retrieve documents). Finally, during the final answer generation phase (generate answer), the system utilizes the LLaMA 3.1 model accessed through the Groq API infrastructure. The Groq API integration was specifically chosen to enable the system to deliver contextual and accurate answers with relatively low latency to users.

The entire query and answer processing flow in this pipeline is executed through an API that acts as a bridge between the chatbot backend module and the web interface. This API is designed to receive text inputs from users, process them through the RAG pipeline, and return responses as text answers. Furthermore, the API is integrated with the website frontend so that users can interact directly with the chatbot through a responsive web interface.

System Evaluation

System evaluation was conducted to assess both answer quality and chatbot performance. For answer quality, RAGAS [23] was used, an open-source library designed specifically to evaluate Retrieval-Augmented Generation (RAG) systems. This study focused its evaluation on three primary metrics: Faithfulness (factual accuracy of the answer), Context Precision (relevance of the retrieved documents), and Context Recall (information retrieval capability). The choice of these three metrics was based on the characteristics of academic information services that prioritize factual accuracy. The Answer Relevance metric was excluded from this evaluation due to rate limits encountered on the judge model when using the Groq API.

Meanwhile, system performance evaluation was conducted by measuring the average response latency and system stability. Latency was calculated from the time a question was sent until the answer was received by the user. Stability was tested by running a load simulation using simple code to determine if the chatbot remains stable when accessed by multiple users simultaneously.

3. Results and Discussion

This section explains the research findings and discusses the developed Retrieval-Augmented Generation (RAG)-based academic information chatbot. The discussion is focused on the results of system implementation, chatbot answer quality testing, and performance evaluation using relevant metrics. Additionally, this chapter analyzes the interpretation of the evaluation results to assess the system's effectiveness in providing accurate and contextual academic information for students of the Informatics Study Program at Universitas Muhammadiyah Sidoarjo.

System Implementation Results

The implementation result of this research is a developed academic information chatbot based on Retrieval-Augmented Generation (RAG). This chatbot is designed to assist users in obtaining academic information quickly and contextually. The chatbot interface is presented as an interactive component on a responsive web page, where users can type questions into a chat column and receive real-time replies. The implementation of the chatbot interface can be seen in Figure 3.1.

Figure 3.1. Chatbot Interface Implementation on the Website.

Based on the testing scenarios, the system successfully executed the LangGraph workflow. When a query is classified as rag_chat, the system searches the ChromaDB vector database. The system retrieves not only text but also supports accompanying metadata such as direct links (direct_link) stored in the JSON format. This ensures that the answers provided are not only descriptive but also practical. Examples of chatbot responses to academic queries are presented in Table 3.1.

Table 3.1. Examples of Chatbot Response Results.

Question	Answer
What is the procedure for requesting a certificate of active enrollment?	Hello! For the procedure of applying for a certificate of active enrollment, you can submit a request through the Informatics Online Letter Administration Service. Here are the steps: 1. Open the Online Letter Form. 2. Select the type of letter you need, which is the Active Student Letter. 3. Fill out the form with the required data, such as name, NIM, and reason for application. 4. Upload necessary documents, if any. 5. Click the "Submit" button to send your request. After you apply, the administration team will process and send your active enrollment letter via email or through the online system. If you need further assistance or have other questions, feel free to ask!
When is the even semester practicum schedule for Informatics students?	Hello! I'm happy to help answer your question about the even semester practicum schedule for Informatics students. To find out the even semester practicum schedule, you can download the relevant practicum schedule via the following links: - Jarkom Practicum Schedule: [Download Computer Networks Practicum Schedule] - Algorithm Practicum Schedule: [Download Algorithm and Data Structures Practicum Schedule] - Database Practicum Schedule: [Download Database Practicum Schedule] Please download the relevant schedule to find out more information about the even semester practicum schedule.
What is the vision and mission of the Informatics Study Program?	Hello! I'm happy to help answer your question about the vision and mission of the Informatics Study Program. Vision: The Informatics Study Program of UMSIDA aims to produce graduates who are professional, excellent, innovative, and competitive in software engineering and intelligent systems that are adaptive to science and technology developments based on Islamic values for the welfare of the ASEAN community by the year 2038. Mission: The mission of the UMSIDA Informatics Study Program consists of 5 points, namely: 1. Excellent & innovative education based on Islamic values. 2. Science and technology research for community welfare. 3. Internationally reputable community service. 4. Sustainable cooperation. 5. Professional governance & student development. Hope this information helps you understand the vision and mission of the UMSIDA Informatics Study Program!

Answer Quality Evaluation

Answer quality evaluation was conducted using the RAGAS framework to measure how accurate and relevant the answers generated by the LLaMA 3.1 model were. Testing was performed on a test dataset consisting of questions surrounding academic, administrative, and student affairs. The evaluation focused on Faithfulness, Context Precision, and Context Recall scores.

Table 3.2. RAGAS Evaluation Results.

Evaluation Metric	Value
Context Precision	0.86
Faithfulness	0.77
Context Recall	0.75

Based on Table 3.2, the system achieved its highest performance in the Context Precision aspect with a score of 0.86. This value indicates that the system's retriever is highly effective in filtering documents, where most of the retrieved text chunks have high relevance to the user's query. Meanwhile, the Faithfulness score of 0.77 shows a fairly good level of consistency between the model's answers and the source documents, although some variations in the LLaMA 3.1 model's answers still occur. The Context Recall score of 0.75 signifies that the system is capable of filtering the necessary information, a figure considered adequate for the initial implementation stage of an information service system.

System Performance Evaluation

Performance evaluation was conducted to measure the resilience and response speed of the system. Testing was carried out by simulating a load of 20 concurrent users simultaneously with a total of 100 requests. The system performance statistics can be seen in Table 3.3.

Table 3.3. System Performance Statistics.

Parameter	Test Results
Total Requests	100
Success Rate	94% (94 Success, 6 Failed)
Throughput	4.14 requests/second
Average Latency	2.11 seconds
P95 (95% of Users)	2.27 seconds
Max Latency	15.99 seconds

The test results showed a success rate of 94%. The average response latency was recorded at 2.11 seconds. Meanwhile, the P95 value (95th Percentile) stood at 2.27 seconds, indicating that 95% of user requests were served in less than 2.3 seconds. To analyze latency stability, a mapping of response times against the request sequence was conducted, as presented in Figure 3.2.

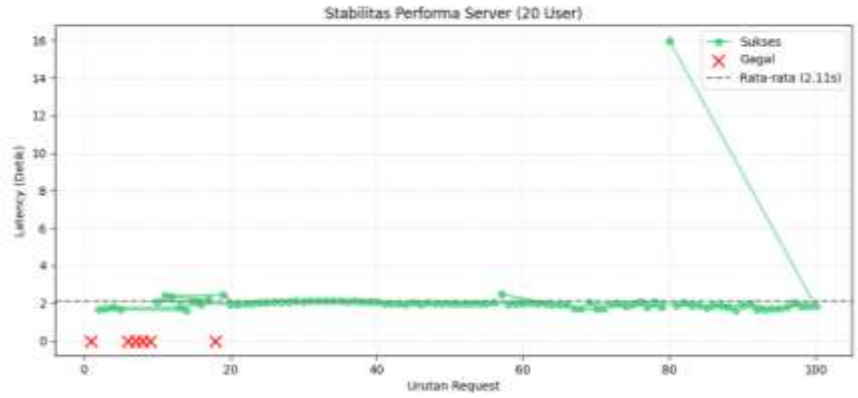

Figure 3.2. System Performance Stability Graph.

Based on the stability graph in Figure 3.2, the majority of requests (green dots) sit on a flat line around the 2-second mark. Meanwhile, request failures (red X marks) occurred during the initial phase of testing (requests 1-19), indicating constraints when the system required initialization time to load the model upon receiving a sudden traffic spike. Additionally, a latency spike was detected at the 80th request, where latency jumped to 15.99 seconds, likely caused by queue mechanisms on the Groq API service provider's side. Despite these constraints, the system was able to recover performance quickly, returning to its normal average latency of 2 seconds.

Discussion

Based on the evaluations conducted, the development of the chatbot using the Retrieval-Augmented Generation (RAG) method with the LLaMA 3.1 model via the Groq API shows optimal performance for academic information services.

Chatbot System Implementation

The use of the LangGraph architecture allows the query processing flow to run structurally, starting from query classification, document retrieval, to final answer generation using the LLaMA 3.1 model. The success of this implementation is evident from the system's ability to classify queries that require document retrieval (rag_chat) from general questions that can be answered directly by the model. Furthermore, the integration of the ChromaDB vector database enables the system to retrieve relevant text chunks along with supporting metadata, such as direct links to information sources. This makes the chatbot not only informative but also practical, as users can directly access the official sources cited.

Answer Quality

The Context Precision score of 0.86 indicates that the retrieval system is capable of choosing relevant documents with a high level of accuracy. This shows that preprocessing data into a structured JSON format and using the nomic-embed-text embedding model contributed significantly to the contextual accuracy provided to the LLM. The Faithfulness score of 0.77 shows that the answers generated by the LLaMA 3.1 model are largely consistent with the source documents used in the retrieval process. Although text variations still occur due to the generative nature of LLMs, the generated responses remain

within the correct boundaries of academic information. Meanwhile, the Context Recall score of 0.75 indicates that the system has been able to capture most of the information required by users, though there is still room to expand the scope of retrieved documents.

System Stability and Performance

The load testing results demonstrate that the system has a success rate of 94% with an average response latency of 2.11 seconds. The P95 value of 2.27 seconds indicates that the vast majority of user requests can be served within a relatively short and consistent timeframe. This result proves that using the Groq API as the LLM inference infrastructure successfully minimizes the latency that frequently limits RAG-based chatbot systems. The maximum latency spike of 15.99 seconds and the request failures during the initial testing phase highlight limitations in the system initialization process and potential queuing within the Groq API service. Nonetheless, the system is capable of returning to a stable condition rapidly, meaning that overall, the chatbot is deemed reliable enough to be used as a web-based academic information service.

In general, direct integration into the study program's website using the LangGraph architecture proves to be superior in terms of accessibility compared to solutions based on Telegram or separate third-party applications discussed in prior studies.

4. Conclusion

This study concludes that the implementation of a Retrieval-Augmented Generation (RAG)-based academic chatbot using the LLaMA 3.1 model via the Groq API successfully increases the efficiency of student information access compared to static websites. Based on the evaluation, the system is proven capable of presenting contextually accurate answers, achieving a Context Precision score of 0.86 and a Faithfulness score of 0.77. Additionally, the average response latency of 2.11 seconds is considered highly adequate for general information retrieval needs, ensuring users obtain answers without long wait times. The integration of the LangGraph architecture and ChromaDB vector database in this system yields actionable responses by including relevant data source links. Based on this implementation, future research is recommended to expand the system's scope to the faculty or university level at Universitas Muhammadiyah Sidoarjo to provide a more comprehensive information service. This expansion can be achieved by integrating academic databases from various study programs through broader automated web scraping techniques.

REFERENCES

[1] A. A. Zulfa, T. Ibrahim, and O. Arifudin, "Peran Sistem Informasi Akademik Berbasis Web Dalam Upaya Meningkatkan Efektivitas Dan Efisiensi Pengelolaan Akademik Di Perguruan Tinggi," *J. Tahsinia*, vol. 6, no. 1, pp. 115–134, 2025.

[2] S. Naikar and C. Indraji, "AI-Driven Innovation in Information Retrieval: Transforming Library Operations and User Engagement," *Int. Conf. Glob. Perspect. Open Sci. Bridg. Res. Access Innov.*, no. May, pp. 1–7, 2025, [Online]. Available: http://papers.ssrn.com/abstract_id=5261229

[3] L. Costaner, P. Studi, T. Informatika, F. Ilmu, K. Universitas, and L. Kuning, "Aplikasi Chatbot untuk Layanan Informasi dan Akademik Kampus Berbasis Artificial Intelligence Markup Language (AIML)," pp. 291–300, 2020.

[4] F. Purwani *et al.*, "IMPLEMENTASI APLIKASI CHATBOT SEBAGAI MEDIA INFORMASI PADA PENGEMBANGAN SISTEM AKADEMIK UNIVERSITAS ISLAM NEGERI (UIN) RADEN FATAH PALEMBANG MENGGUNAKAN ARTIFICIAL INTELLIGENCE MARKUP LANGUAGE," vol. 1, no. 4, pp. 1–7, 2024.

[5] M. F. Ajiz, M. Faza, S. Ramadan, H. D. Mutia, P. D. Yanuari, and I. Pendahuluan, "Pengembangan Aplikasi Chatbot Informasi Akademik Berbasis Web Menggunakan Metode Artificial Intelligence Markup Language (AIML)," vol. 15, no. 2, 2023.

[6] N. Cahyono, "Perancangan Sistem Informasi Penyewaan Alat Outdoor Berbasis Web menggunakan Metode Waterfall pada InOutdoors Rental Sidoarjo," no. 1, pp. 1–23, 2024.

[7] A. Alvin, R. Robet, and A. Tarigan, "Implementasi chatbot Otomatis Akademik Berbasis Web Menggunakan LLM dan Rule-Based System Studi Kasus : STMIK Time Implementation of Web-Based Academic Automated chatbot Using LLM and Rule-Based System Case Study : STMIK Time," no. 3, pp. 651–665, 2025, doi: 10.26798/jiko.v9i3.2209.
[8] M. I. T. Khaqiqi and N. H. Harani, "Peningkatan Kinerja Chatbot NLP Asisten : Tinjauan Literatur tentang Metode dan Akurasi dalam Aplikasi Berbasis Percakapan," vol. 05, no. 01, pp. 50–59, 2024.
[9] D. Apriliani, S. F. Handayani, I. T. Saputra, T. Informatika, and P. H. Bersama, "Implementasi Natural Language Processing (NLP) Dalam Pengembangan Aplikasi Chatbot Pada SMK YPE Nusantara Slawi," vol. 22, no. 4, pp. 1037–1047, 2023.
[10] S. Abedu and A. Abdellatif, "LLM-Based Chatbots for Mining Software Repositories : Challenges and Opportunities," no. June 2024, 2025, doi: 10.1145/3661167.3661218.
[11] C. Y. Kim, C. P. Lee, and B. Mutlu, "Understanding Large-Language Model (LLM)-powered Human-Robot Interaction," *ACM/IEEE Int. Conf. Human-Robot Interact.*, no. Llm, pp. 371–380, 2024, doi: 10.1145/3610977.3634966.
[12] S. Vidivelli, M. Ramachandran, and A. Dharunbalaji, "Efficiency-Driven Custom Chatbot Development: Unleashing LangChain, RAG, and Performance-Optimized LLM Fusion," *Comput. Mater. Contin.*, vol. 80, no. 2, pp. 2423–2442, 2024, doi: 10.32604/cmc.2024.054360.
[13] Milasanti D, "SISTEM CHATBOT BERBASIS LARGE LANGUAGE MODEL (LLM) DANRETRIEVAL AUGMENTED GENERATION (RAG) PADA ARTIKELILMIAH GARUDA KEMDIKBUD," 2024.
[14] Nu'aeni H, "PENGEMBANGAN CHATBOT UNTUK SISTEM INFORMASIPERPUSTAKAAN DIGITAL (DIGILIB) MENGGUNAKANRETRIEVAL-AUGMENTED GENERATION (RAG)," 2025.
[15] L. R. Hidayat, G. Pasek, S. Wijaya, and R. Dwiyansaputra, "Optimalisasi Layanan Sistem Informasi Mahasiswa dengan Integrasi Telegram : Chatbot Retrieval-Augmented-Generation berbasis Large Language Model (Optimization of Student Information System Services with Telegram Integration : Chatbot Retrieval-Augmented Generation based on Large Language Model)," 2021. [Online]. Available: http://jtika.if.unram.ac.id/index.php/JTIKA/
[16] M. Grusky, "Rogue Scores," vol. 1, pp. 1914–1934, 2023.
[17] U. States, "Towards Seamless User Query to REST API Conversion," no. October 2024, 2025, doi: 10.1145/3627673.3680275.
[18] I. Mohr, D. J. Williams, B. Wang, and H. Xiao, "L ATE C HUNKING : C ONTEXTUAL C HUNK E MBED -," pp. 1–14, 2024.
[19] M. A. Rosid, D. O. Siahaan, and A. Saikhu, "Sarcasm Detection in Indonesian-English Code-Mixed Text Using Multihead Attention-Based Convolutional and Bi-Directional GRU," no. June, pp. 137063–137079, 2024.
[20] Z. Nussbaum, J. X. Morris, and B. Duderstadt, "Nomic Embed : Training a Reproducible Long Context Text Embedder," vol. 002, pp. 1–17, 2025.
[21] T. Chatbot, "Implementation of Retrieval-Augmented Generation (RAG) and Large Language Models (LLM) for a Document and Tabular-Based Chatbot System," pp. 19–23, 2025.
[22] J. Wang and Z. Duan, "Agent AI with LangGraph: A Modular Framework for Enhancing Machine Translation Using Large Language Models," 2024, [Online]. Available: http://arxiv.org/abs/2412.03801
[23] S. Es, J. James, L. Espinosa-anke, and S. Schockaert, "RAGA S : Automated Evaluation of Retrieval Augmented Generation," pp. 150–158, 2024.