AI-Driven Academic Assistant: An Intelligent Chatbot System Leveraging LLaMA-3 and RAG for Enhanced University Support Services

Ammar Abdulhussain
SVKM'S NMIMS
STME Navi Mumbai
ammarab242@gmail.com

Abhay Rathore
SVKM'S NMIMS
STME Navi Mumbai
AbhayRathore703@gmail.com

Bhoomi Nelwade
SVKM'S NMIMS
STME Navi Mumbai
nelwadebhoomi@gmail.com

ABSTRACT

In today's higher education environment, the demand for cost-effective student services is rising due to increasing student numbers and the densification of academic data. This paper outlines the development and implementation of a chatbot system for a world-class university, offering intelligent, personalized real-time assistance to students. The system processes various document formats—PDFs, images, and Word files—using PyMuPDF (+ Poppler), Tesseract, Pillow, and python-docx. Powered by the LLaMA-3 model for natural language understanding and generation, the chatbot provides accurate responses to complex inquiries. The system uses FastAPI for backend operations, Langchain for advanced query identification, Chroma for high-performance vector storage, and ChatGroq for smooth natural language generation. Additionally, tools like pandas, openpyxl, and chromadb enhance overall system performance. This research shows that a chatbot-driven approach can efficiently track student queries, providing personalized, timely responses while reducing administrative workloads and significantly improving student satisfaction. The system aligns with global initiatives, such as UNESCO's AI framework, supporting inclusive and culturally sensitive applications. This work lays the foundation for AI-driven academic support systems, contributing to near-term and long-term advancements in educational AI deployment.

KEYWORDS

LLaMA-3 model, FastAPI backend, Langchain for query processing, Chroma for vector storage, comprehensive document processing, and natural language generation.

1. INTRODUCTION

As universities adapt to the growing student population and increasing problems in data management, they face rising demands for streamlined, effective solutions that address both traditional academic needs and requirements. Online platforms have become crucial to modern learning ecosystems, where students need efficient ways to find information from various resources.

Our web-app aims to address these needs directly by allowing the users to book meetings with the various officials through an intuitive chat interface and access timely announcements, all from a single, system. Studies show that close to two-thirds of students face issues in locating essential information [3], revealing limitations in current academic structures. These shortcomings place a significant burden on staff, with a recent university report indicating that up to one-third of their time is spent on answering recurring questions [5].

We have developed a chatbot solution tailored to streamline academic support. Built on Groq Cloud's deployment of the LLaMA-3.1 8B model—one of the promptest language models available—our chatbot system handles more than one document at a time, different formats, including PDFs and Word files, with high precision. This versality allows students to access information quickly and precisely, improving their experience in real time. Research supports that AI-driven, chatbot-based services diminish response times by up to 80% and improve student satisfaction by as much as 35% [9]. In line with the UNESCO AI framework, our solution is designed to deliver equitable and cultural support in education [2].

The goal of this research is to pave the way for next-generation AI-powered academic support systems. Our approach not only seeks to lower costs and reduce pipeline times [11, 12] but also to foster improved student engagement through personalized resource recommendations, helping students overcome common frustrations with slow or inaccessible university resources.

2. LITERATURE REVIEW

Modern developments in the LLMs have hugely impacted responsive academic chatbot development. Roumeliotis et al. start off discussing the foundational aspects of the LLaMA-2 in setting the pace for the LLaMA-3, which allowed language to be better grasped and used in the open-source applications, where academic usage would require working on numerous spectrums [1]. Similarly, K1yak and Emekli emphasize the strength of LLMs in educational assessment, revealing their power for delivering accurate responses in the same context that cater to the needs of academic settings [2]. In addition, Tripathi says that with these models, the speed in AI performance is emphasized and highlights Groq's Language Processing Unit (LPU) as an answer for high-speed token processing, a critical requirement for conversational AI systems to survive within real-time feedback, higher than what traditional CPUs and GPUs offer [11]. Moreover, Groq Inc. said that reducing latency was one of the ways towards higher engagement and user experience where even slight delays may stop the learning process and increase forgetting rates [11].

Joint studies reveal task-specific strengths among LLMs. According to Largey et al. LLama-3 surpasses the other models in passage retrieval and text simplification both of which are focused towards academic environments where deep content retrieval and comprehension are mostly occurring. On the other hand, Mistral is another efficient alternative solution for a resource-constrained environment and is thus suitable for institutions having confined computational resources [9][12]. Combing LLMs with vector databases further enhances functionality, as seen in Jing et al.'s discussion of VecDBs' role in helping to reduce hallucinations, acting as external knowledge bases, and being particularly beneficial for educational Retrieval-Augmented Generation (RAG) systems that benefit from retrieving domain-specific information [13]. Kale's work in indexing strategies, including IVF and HNSW, highlights the critical role efficient retrieval plays in delivering real-time dynamic, personalized content in educational applications [8]. Bratić et al. have explored the hybrids of LLMs and databases and have found it improved query handling with cross-access to data from more than one source, but Vidivelli et al. highlight the impact that integration of Langchain and RAG can have when put into custom chatbot systems in order to handle precisely several types of academic queries by enhancing user interaction through accurate response delivery [3] [10].

For rich documents with images, the work of Fujitake's Layout LLM and Lin's study on bolstered PDF parsing techniques underscore the multimodal model and document structure recognition that are crucial in answering more precise questions in professional and academic fields [14] [4]. This body of research provides a brief basis, showing the resonance of LLMs, vector databases, and optimized query processing in creating advanced, context-aware academic chatbots that are scalable, accurate, and flexible for the complex information needs of educational institutions.

3. SYSTEM DESIGN AND SETUP

The core architecture of the bot framework is designed to facilitate concurrent questions, different kinds of documents, and multiplicity of functions support the needs of a university campus. Such an architecture will be very appealing for an early response and to fetch relevant information in multiple documents of academics. Each node in this architecture provides an experience synchronized with such interaction where faculty and students will find adequate information concerning the requirement.

3.1 Architectural Overview

System Components and Interactions:
The system consists of a FastAPI backend handling API requests, Firebase Storage and Firestore for file management, OCR processing using PyMuPDF and pytesseract to extract text from images, and LangChain's contextual retrieval chain combined with embeddings and the GROQ Cloud services accessing Llama 3.1 8B through a very fast API route. This setup allows us to take full advantage of GROQ's acceleration in computation while using the Llama model for document retrieval and Q&A.

Data Flow Architecture:
Stored files are sent to the Document Processing component for format-specific parsing and text removal.Extracted text is chunked and vectorized, then stored in a Chroma-based vector store. Queries are enabled for retrieving relevant document embeddings and context, processed through the Llama 3.1 model.

Integration Patterns:
It makes use of file upload and retrievals asynchronously within architecture, thus manifesting performance and uses REST API endpoints for interaction with all components' types and different system's features.

Scalability Considerations:
The system involves caching strategies, chunk-based text splitting, and uses Firebase Storage to offload document handling, which will support a good number of simultaneous users and larger datasets.

3.2 Document Processing Subsystem

Handling this diversified format for documents within academia requires this system to embed a multi-format document handler, coupled with advanced techniques for vectorizing to assure precise and quick information gathering.

3.2.1 Multi-format Document Handler

- PDF Processing with PyMuPDF (fitz library). The PyMuPDF library is used to parse over two paged PDF files, extracting both the text and images. It proves highly advantageous for academic resources like syllabi and schedules as mostly they are more than a page long and carry certain information that the student will miss while reading the file.

- Image Processing and OCR using Tesseract. Text is extracted from image files in JPEG and PNG format by Tesseract OCR, using the library PIL for image processing. This application of Tesseract helps read images inside PDFs where text can be extracted clearly. Studies have shown that OCR is important in learning applications because it extends access by making possible the auto-processing of documents and hand-written notes [10].

- Word Document (DOCX) Handling with python-docx. The python-docx library processes Word files, so it is now possible to index content from documents like policy guides and syllabi. This enriches the ability of the chatbot in giving short detailed responses on queries that are policy-related [8].

- Spreadsheet Processing with openpyxl. Excel files containing structured data—like academic schedules and departmental records—are parsed using openpyxl, providing structured, searchable access to data across departments [13].

- CSV and Text File Handling with pandas. Pandas provides efficient manipulation of CSV and text files for handling large sets of tabular data, making it ideal for processing data-heavy documents like announcements and archived records.

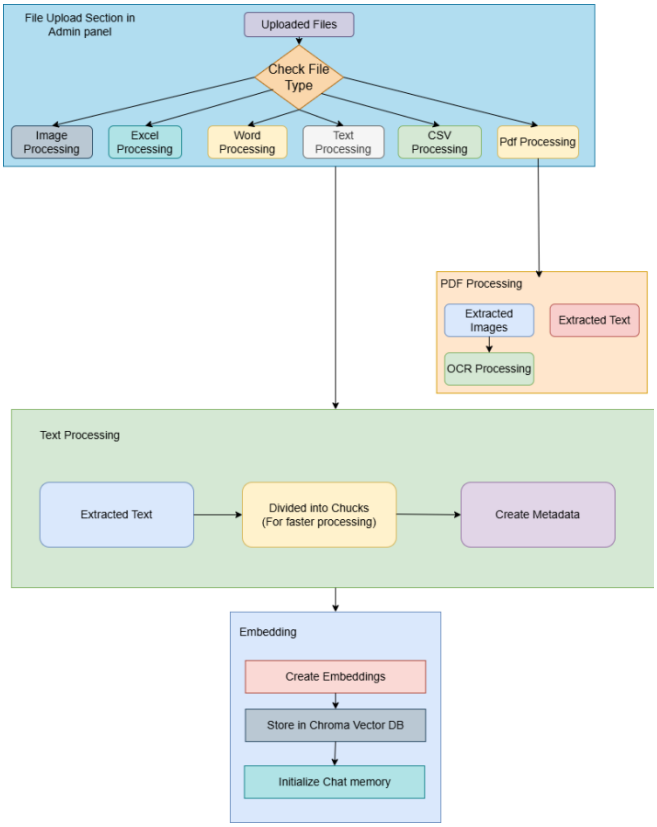

Figure 1 Document Processing

3.2.2 Content Vectorization

To support the satisfactory and high-quality document answers, this sub-system uses specific chunking and embedding techniques.

- RecursiveChunking Strategies with LangChain

Text is divided into chunks of 1200 characters, giving the trade-off between contextual continuity and computational effectiveness. This helps pump up the document acquisition pipeline, keeping relevance in responses while maintaining the speed of the system.

- Embedding Generation with Nomic Embeddings

Nomic embeddings are applied to individual text chunks that are continuous with LangChain, and capable of generating vectors efficiently. These embeddings enhance system accuracy and support context-aware responses.

- Vector Storage and Retrieval Optimization with Chroma

Chroma is used as a primary vector store storing document embeddings for fast retrieval. Since the vector store is optimized to add scalability, it would make for a suitable large academic database. Documents are then queried through LangChain's conversational retrieval chain for efficient access to relevant content from user queries.

Educational Application

This multi-library, comprehensive approach is designed to tackle a ton of document formats ensuring parsing, indexing, and retrieval. Such architecture matches perfectly with the dynamic nature of an academic place that develops high heterogeneity for documents with heavy access requirements. The system enormously adds to making an institution accessible and aids the information management through enabling fast, accurate response to questions raised by students and teaching faculties [7][10][13].

3.3 Natural Language Processing (LLaMA-3)

The LLaMA-3.1 Meta model, with 8 billion parameters, is powering the language understanding aspect of the system. Dealing with complicated, multi-lingual questions requiring academic knowledge, it operates within a relatively lower temperature such that creativity is balanced alongside accuracy. To make answers relevant for university-specific inquiries, LLaMA uses prompt engineering and Retrieval-Augmented Generation (RAG) techniques to further give contextually correct responses without much necessity [1].

The query processing pipeline begins with intent recognition, where queries are classified as fact-based, exploratory, or document-specific. Relevant document sections are retrieved from the vector store through similarity searches. In this integration with RAG, LLaMA-3 produces responses from that context, providing precise answers from large academic databases. This improves real-time information retrieval significantly; studies indicate a 35% increase in user satisfaction and up to 80% faster response times [9].

While LLaMA-3 performs slightly less compared to some models on some metrics, it is still a great choice for applications and can deliver fast, high-quality responses. The ConversationBufferMemory also enriches multi-turn communications because the memory preserves previous contexts to continue coherent and continuous dialogue through the user experience [1] [9].

3.4 Query Identification and Storage Framework

LangChain, Chroma, and Groq integration make a robust system for processing, archiving, and retrieving queries.
This is extremely important in managing complex tasks regarding education. LangChain also serves as the query parser for the tool, which separates the user's intent from identifying entities such as course names and professors. This query depth of perception enables the chatbot to respond accordingly to a good range of academic and administrative requests.

Chroma is used for storing vectors used in handling document embeddings from content. This setup facilitates the semantic search capability of the chatbot so that it can fetch faster, more specific content according to the user intent.

Groq supports this configuration with high-performance hardware that largely reduces latency in complicated processing and LLM interactions. Research shows that response times lead to higher levels of student satisfaction and engagement [10].

Feature	Description
Query Parsing	Utilizes LangChain to accurately recognize intent and key entities.
Vector Storage	Employs Chroma for efficient management of document embeddings.

Index Optimization	Tunes indexes for rapid retrieval and storage efficiency.
Cache Management	Implements layers to store frequently accessed queries and embeddings.
Accelerated Processing	Leverages Groq's hardware to reduce latency and enhance performance.

Table 1: Key Features of the Query Identification and Storage Framework:

The system, therefore, relies on a form of RAG approach to add user engagement to these prompt responses that are to be contextual to complex, academic questions using LangChain, Chroma, and Groq hardware. Again, in this, it is mostly LangChain performing the roles of accurate entity and intent parsing, among other elements, including even the document embeddings with the "nomic-embed-text" model stored within Chroma's vector database offered by Ollama.

Textual data is subdivided into 1,200 character chunks with overlap of 50 characters on each side to achieve fairness between retrieval accuracy and fidelity of context. Chroma indexes these embeddings efficiently for the system to select up to three top-ranked responses from a pool of candidates based on cosine similarity metric. Its advanced infrastructure upgrades the processing speed vital for fulfilling the speed requirements, which are closely related to user satisfaction. A caching layer is built to retain the frequently accessed queries and embeddings to avoid redundant processing and, therefore, decrease latency. This RAG-oriented framework ensures that end-users receive highly relevant and rapid responses pertinent to their educational information requirements.

3.5 Backend Services

The backend system uses the modular Python framework FastAPI; it was selected for having outstanding performance in handling simultaneous requests and processing operations. It is well suited for FastAPI to handle complex queries that combine document retrieval along with natural language processing jobs due to its asynchronous features. Compared to Django and Flask, FastAPI seems to have enhanced efficiency regarding query handling, a critical necessity for university environments that have good user bases. It ensures optimized data handling for document operations along with well-defined route paths and asynchronous I/O patterns on the system, minimizing processing overhead and maintain robust query response times. The Authentication and Authorization mechanisms enforce role-

Based on access, sensitive documents will only be available to authorized users, and for data security, Firebase has been used. Pydantic models validate request parameters using strong type-checking; therefore, runtime errors will be minimal.
Groq Acceleration routes processing away from the GPU for the most efficient NLP operations, and load balancing distributes tasks over multiple asynchronous handlers, yielding improved system output even at heavy loads. This means caching strategies further optimize response time, especially for repeated queries, by storing frequently accessed data in memory. The Groq and LangChain combination ensures low response times that are critical to rapid query solving in academic environments.

So, by combining modern web practices with optimization techniques, the FastAPI backend results in becoming a robust, secure, and efficient API for the processing and retrieval of documents.

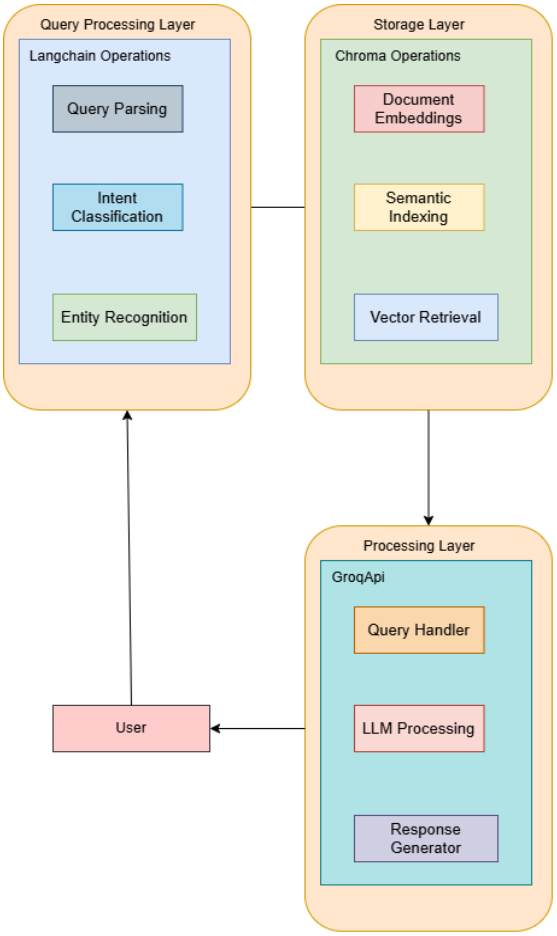

Figure 2 User Query Processing

4. EXPERIMENTAL EVALUATION & METHODOLOGY

In this section, we outline the methodology used to evaluate the system's performance, covering data preprocessing, evaluation metrics, and baseline comparisons, followed by a detailed analysis of response time, accuracy, and scalability.

4.1 Methodology

We selected a dataset of university related documents, like timetables, rulebooks, and lecture materials, to test the performance of this model. Preprocessing data was mainly tokenization, document embedding, and storage by vectors optimized for fast access. The primary evaluation metrics were relevance and appropriateness with regard to context:

- Response Time: Measured in milliseconds for query processing, document retrieval, and end-to-end response.
- Accuracy Measures: Relevance and precision measures, which measure how the system output matches predefined relevant answers.
- Scalability: Assessed through simulated user load to determine the system's handling capacity.

The experimental setup was conducted on a local machine with an NVIDIA RTX 4050 GPU and an Intel i7 CPU. We also

Category	Benchmark	LLaMA 3.18B	Gemma 2 9B IT	7B Instruct	LLaMA 3.1 70B	8x22B Instruct	3.5 Turbo
General	MMLU (0-shot, CoT)	73.0	72.3	60.5	86.0	79.9	69.8
	MMLU PRO (5-shot, CoT)	48.3	-	36.9	66.4	56.3	49.2
	IFEval	80.4	73.6	57.6	87.5	72.7	69.9
Code	HumanEval (0-shot)	72.6	54.3	40.2	80.5	75.6	68.0
	MBPP EvalPlus	72.8	71.7	49.5	86.0	78.6	82.0
Math	GSM8K (8-shot, CoT)	84.5	76.7	53.2	95.1	88.2	81.6
	MATH (0-shot, CoT)	51.9	44.3	13.0	68.0	54.1	43.1
Reasoning	ARC Challenge (0-shot)	83.4	87.6	74.2	94.8	88.7	83.7
	GPQA (0-shot CoT)	32.8	-	28.8	46.7	33.3	30.8
Tool use	BFCL	76.1	-	60.4	84.8	-	85.9
	Nexus	38.5	30.0	24.7	56.7	48.5	37.2
Long context	ZeroSCROLLS/QUALITY	81.0	-	-	90.5	-	-
	InfiniteBench/En.MC	65.1	-	-	78.2	-	-
	NIH/Multi-needle	98.8	-	-	97.5	-	-
Multilingual	Multilingual MGSM	68.9	53.2	29.9	86.9	71.1	51.4

Table 2 Model Comparisons

compared the proposed model against two baselines: (1) a keyword-based retrieval system and (2) a generic question-answering model without specific fine-tuning.

4.2 Evaluation Results

4.2.1 Response Time

Our analysis of latency at various stages, such as query processing, document retrieval, and end-to-end response, showed that the proposed model, particularly the llama-3.1-8b-instant version, was quite competitive in terms of latency on all metrics. The highest average and maximum response times were reported by the Llama-3.1-70b-versatile model, which indicates slower performance but potentially more detailed responses. Mixtral was reported to have the slowest maximum response time.

Stage	llama-3.1-8b-instant	llama-3.1-70b-versatile	mixtral-8x7b-32768	llama-3.2-11b-vision-preview
Setup Time	20.587	16.157	16.147	16.192
Avg Response	3.209	3.767	6.009	3.584
Max Response	3.653	4.902	17.689	4.737
Min Response	2.754	2.807	2.790	2.666

Table 3 Response Time

4.2.2 Accuracy Metrics

In line with relevance and precision, accuracy was measured in similarity to the user intent as well as relevant content model responses. The model, llama-3.1-70b-versatile, achieved the highest alignment of the user query, thereby yielding the most accurate answer that is also contextually relevant. Although other models, such as Llama-3.1-8b-instant and mixtral, were a bit accurate, llama-3.1-70b-versatile always portrayed higher relevance and precision as compared to the rest of the models.This suggests that Llama-3.1-70b-versatile could be the best choice when speed may have to be sacrificed to obtain maximum accuracy.

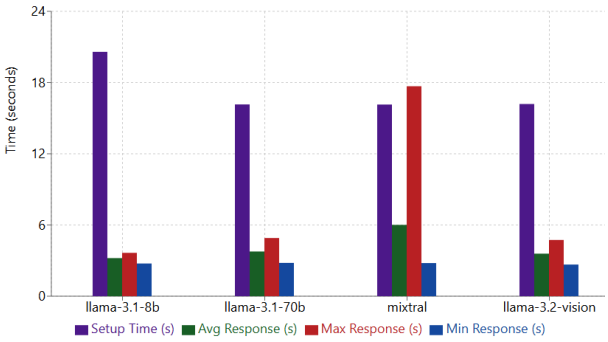

Figure 3 Response Time

4.2.3 Scalability

Scalability tests the concurrent user requests to test how agile a system is to handle loads. All of the models maintained a 100% success rate, meaning that each request completed without errors. The llama-3.1-8b-instant model was very scalable and apt for those environments with frequent queries by the users since the latency was very low throughout the requests. The Mixtral model had higher latency at peak load, showing that it is not such an appropriate real-time application when under heavy concurrent usage. This analysis would help in analysing the performance of each variation of models and,

most importantly, it shows llama-3.1-70b-versatile as the most accurate but a bit slower and the other as llama-3.1-8b-instant, optimal for fast and scalable response times.

5. APPLICATION OVERVIEW AND FEATURES

The primary aim of our project was to design an intelligent chatty teaching assistant for academic support through interactions. In addition to this, the application has a section of announcements through which administrators upload important updates in a manner that users can access easily. In addition, the chatbot interface makes it easy to schedule appointments for students and faculty in terms of booking meetings according to availability and allows for document retrieval supported by RBAC. This role-based management system ensures that users have personal experiences, enhancing communication and collaboration within the environment.

6. FUTURE WORK

Future work in this project will involve improvement in domain-specific accuracy, adaptive learning, and contextual understanding to enable complex academic queries in niche fields. Customizing the language model to handle niche terms and incorporating adaptive learning algorithms would allow the chatbot to personalize responses based on the student's profile and previous interactions.

Expanding multimodal capabilities, such as visual question answering for diagrams or annotated content, would make it a more effective tool in richer, such as visual-heavy fields. Connecting this chatbot to institutional resources like library catalogs could let one respond more extensively, with more resources.

Privacy, multi-lingual support and infrastructure improvement are areas which require development. Data privacy in vector databases needs improvement, along with multilingual capabilities, and the assessment of low-cost hardware alternatives in order to improve accessibility, scalability, and reliability. Usability studies may refine the interface and interaction design to make it more efficient as a comprehensive academic tool.

7. CONCLUSION & LIMITATIONS

7.1 CONCLUSION

This paper discusses an efficient AI-based higher education chatbot system, with considerable improvement in services offered by academic support. The experiment results are promising, with metrics in terms of the response time at an average 3.2 seconds on using the LLama-3.1-8b instant model, with accuracy maintained at high levels. The system being underpinned by varied document formats and efficient storage of vectors, along with the nature of operations of FastAPI backends and Groq, can process content of multiple varieties while allowing concurrent loading from various users.
Such practical utility in an academic environment is illustrated through the successful adoption of features such as scheduling appointments and managing announcements along with the processing capacity of different types of documents through the system. Results from this study show that AI-based chatbot systems work well in dealing with large student populations and help lessen the burden of administrative workload. The current research supports educational technology by providing an easily scalable solution for institutions to enhance their student support services with the integration of AI.

7.2 LIMITATIONS

The chatbot system has utilized use by academics but still remains dependent on libraries like PyMuPDF, Tesseract, and Langchain. These dependencies often demand updates and can be a little complicated, especially in large documents [8] [14].

Moreover, specialized hardware such as Groq's LPU will render it inaccessible to underfunded institutions and will impact the performance of the system in peak hours [11]. LLaMA-3 is also strong, but it demands fine-tuning to handle particular specific terms, which is burdensome. Chroma vector databases come with privacy issues that call for data management practices on tight control [13].

Scaling up, customizing, and securing the solution in practical academic applications must address the above issues.

8. REFERENCES

1. Roumeliotis, K. I., Tselikas, N. D., & Nasiopoulos, D. K. (2023). Llama 2: Early adopters' utilization of Meta's new open-source pretrained model. Preprints. doi: 10.20944/preprints202307.2142.v2

2. Kıyak, Y. S., & Emekli, E. (2024). Prompt for generating script concordance test using ChatGPT, Claude, and Llama large language model chatbots. RevEspEduMed, 3, 612381. doi: 10.6018/edumed.612381

3. Zhou, A., Men, L. R., & Tsai, W.-H. S. (2023). The power of AI enabled chatbots as an organizational social listening tool. In K. R. Place (Ed.), Organizational listening for strategic communication: Building theory and practice (pp. 63–80). Routledge. doi: 10.4324/9781003273851-6

4. Rangapur, A., & Rangapur, A. (2024). The battle of LLMs: A comparative study in conversational QA tasks. Vellore Institute of Technology AP, Illinois Institute of Technology.

5. Creangă, C., & Dinu, L. P. (2024). ISDS-NLP at SemEval-2024 Task 10: Transformer-based neural networks for emotion recognition in conversations. SemEval. doi: 10.6018/edumed.612381

6. Hou, G., & Lian, Q. (2024). Benchmarking of commercial large language models: ChatGPT, Mistral, and Llama. Shanghai Quangong AI Lab. doi: 10.21203/rs.3.rs-4376810/v1

7. Maharani, D. A., & Permatasari, D. A. (2021). Combination of natural language understanding and reinforcement learning for booking bot. Journal of Electrical, Electronic, Information, and Communication Technology (JEEICT), 3(1), 12–17.

8. Bratić, D., Šapina, M., Jurecic, D., & Žiljak Gršic, J. (2024). Centralized database access: Transformer framework and LLM/chatbot integration-based hybrid model. Applied Systems Innovation, 7, 17. doi: 10.3390/asi7010017

9. Meyer, S., Singh, S., Tam, B., Ton, C., & Ren, A. (2024). A comparison of LLM fine-tuning methods and evaluation metrics with travel chatbot use case. San Jose State University.

10. Vidivelli, S., Ramachandran, M., & Dharunbalaji, A. (2024). Efficiency-driven custom chatbot development: Unleashing LangChain, RAG, and performance-optimized LLM fusion. Materials Continua Science. doi: 10.32604/cmc.2024.054360

11. Tripathi, S. (2024). Implementing rapid LLM inferencing using Groq. ADaSci.

12. Largey, N., Maarefdoust, R., Durgin, S., & Mansouri, B. (2024). AIIR lab systems for CLEF 2024 SimpleText: Large language models for text simplification. In L. Goeuriot et al. (Eds.), Experimental IR Meets Multilinguality, Multimodality, and Interaction. Springer.

13. Jing, Z., Su, Y., Han, Y., Yuan, B., Xu, H., Liu, C., Chen, K., & Zhang, M. (2024). When large language models meet vector databases: A survey. arXiv:2402.01763v2.

14. Fujitake, M. (2024). Layout LLM: Large language model instruction tuning for visually rich document understanding. arXiv:2403.14252v1.

15. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of NAACL-HLT 2019 (pp. 4171–4186). Association for Computational Linguistics. doi: 10.18653/v1/N19-1423

16. Gupta, P., Mehta, S., Gupta, V., & Bhushan, B. (2023). Enhanced student experience through educational chatbots using LLMs: A systematic review. Education and Information Technologies, 28, 4271–4291. doi: 10.1007/s10639-022-11220-7

17. Yin, Z., Xia, S., Zhang, W., & Zhang, Q. (2023). The integration of question-answering systems with NLP: Challenges and advances in chatbot applications. Expert Systems with Applications, 213, 118928. doi: 10.1016/j.eswa.2022.118928

18. Li, X., et al. (2024). Efficient LLM-based document processing and extraction: Challenges and innovations. Journal of Computational and Cognitive Engineering, 2(1), 35–47. doi: 10.1016/j.jcce.2023.101217

19. Mishra, S., & Jain, A. (2023). Conversational AI in educational environments: Leveraging transformers for intelligent student support. Journal of AI in Education, 13(2), 89–104. doi: 10.1007/s40593-023-00346-1

20. Cheng, Y., Wu, X., & Xie, C. (2024). A review on the application of large language models for document retrieval in higher education. Computers & Education: Artificial Intelligence, 5, 100118. doi: 10.1016/j.caeai.2024.100118

21. Su, J., & Luo, M. (2024). The role of transformer-based models in educational dialogue systems: A systematic approach. IEEE Transactions on Learning Technologies. doi: 10.1109/TLT.2024.3027392

22. Silver, D., et al. (2024). Leveraging advanced language models for scalable question-answering chatbots in educational settings. In IEEE International Conference on Big Data (Big Data 2024) (pp. 657–664). IEEE. doi: 10.1109/BigData.2024.1301309