Cureus Journal of Computer Science

Open Access Original Article

# A Modular Retrieval-Augmented Conversational AI Chatbot System with Integrated Recommender Engine Using Local LLMs

**Received** 04/30/2025
**Review began** 05/18/2025
**Review ended** 07/11/2025
**Published** 07/13/2025



**DOI:** https://doi.org/10.7759/s44389-025-06053-3

N. Toyaad K. Reddy [1] , Manas R. Patra [1] , Brojo K. Mishra [1]

1. Computer Science and Engineering, NIST University, Berhampur, IND

**Corresponding authors:** N. Toyaad K. Reddy, toyaad.reddy@nist.edu, Brojo K. Mishra, brojomishra@gmail.com

## Abstract

Recent advancements in large language models (LLMs) and natural language processing (NLP) have enabled more capable conversational AI systems. However, challenges such as hallucinations, limited context retention, and poor personalization persist. This study presents a modular, privacy-preserving conversational AI framework that operates entirely on local infrastructure to address these limitations. The proposed system integrates a Retrieval-Augmented Generation (RAG) architecture using LangChain, with semantic search powered by FAISS vector indexing. Text preprocessing is performed via NLP pipelines, and document embeddings are generated using locally hosted models through the Ollama runtime. At query time, relevant content is retrieved from user-provided documents and web inputs to ground responses in factual context. A hybrid recommender system further enhances personalization by suggesting related content based on semantic similarity and interaction history. The framework includes both a Flask-based API and a command-line interface, offering deployment flexibility. Evaluation demonstrates the system's ability to generate accurate, contextually grounded responses with low computational overhead. By avoiding reliance on cloud services, the framework ensures data privacy without compromising performance. This research contributes a scalable blueprint for secure, responsive conversational AI. Future work will focus on multi-document handling, real-time interaction, and deeper personalization to improve adaptability and user engagement.

**Categories:** AI applications, Recommender Systems, Natural Language Processing (NLP)
**Keywords:** conversational ai, retrieval-augmented generation (rag), natural language processing (nlp), semantic search, local infrastructure, privacy-preserving framework

## Introduction

The past decade has witnessed a transformative evolution in the field of artificial intelligence (AI), especially in the domains of natural language processing (NLP) and machine learning (ML). These advancements have significantly impacted the design and functionality of conversational systems. Early chatbots relied on rigid rule-based architectures and deterministic dialog flows, offering limited user engagement and adaptability. In contrast, modern conversational agents are powered by large-scale transformer-based language models (LLMs) such as generative pre-trained transformer (GPT), Bidirectional Encoder Representations from Transformers (BERT), Large Language Model Meta AI, and Mistral, which enable dynamic, context-aware, and human-like interactions [1-3]. These models have broadened the application spectrum of conversational AI, finding roles in education, healthcare, customer service, and enterprise knowledge management [4,5].

Despite these breakthroughs, current AI-driven conversational agents face several well-documented limitations. Issues such as hallucination of facts, loss of conversational context over time, and insufficient personalization continue to hinder their reliability and user satisfaction in real-world applications [6-8]. Addressing these concerns, retrieval-augmented generation (RAG) has emerged as a promising paradigm. By grounding model responses in externally retrieved, context-relevant documents, RAG enhances factual accuracy and reduces generative drift [2,6,9]. At the same time, recommender systems integral to personalization in digital interfaces have evolved from basic collaborative filtering to sophisticated AI-powered engines that can interpret semantic user intent and behavioral signals [4,10,11].

This research proposes a modular AI chatbot framework that unifies retrieval-based grounding with an intelligent recommender engine, designed for offline deployment without dependency on cloud services. The motivation for this approach stems from increasing demands for data privacy, cost-effective solutions, and domain-specific adaptability, particularly in regulated or infrastructure-limited settings [12,13]. The system utilizes open-source technologies, including locally hosted LLMs via the Ollama runtime, and leverages Facebook AI Similarity Search (FAISS) for high-performance semantic similarity search. The ingestion pipeline supports heterogeneous data sources such as PDFs, DOCX files, and websites, employing natural language preprocessing techniques including tokenization, lemmatization, and stop-word removal [13,14].

**How to cite this article**
Reddy N K, Patra M R, Mishra B K (July 13, 2025) A Modular Retrieval-Augmented Conversational AI Chatbot System with Integrated Recommender Engine Using Local LLMs. Cureus J Comput Sci 2 : es44389-025-06053-3. DOI https://doi.org/10.7759/s44389-025-06053-3

Developed using Python, the system features a Flask-based application programming interface alongside a command-line interface (CLI), supporting flexible integration in diverse environments such as academic repositories, enterprise knowledge bases, or personalized educational platforms. LangChain orchestrates the RAG pipeline, prompt engineering, and conversational memory, while the recommender system employs a hybrid strategy combining frequency-inverse document frequency (TF-IDF) keyword extraction with dense vector ranking [15,16]. This architecture ensures not only responsive question-answering but also intelligent content discovery, enabling users to explore related resources dynamically.

Preliminary evaluations demonstrate high relevance in response generation and low latency performance, even when operating on consumer-grade hardware. The modular design facilitates easy customization enabling the swapping of language models, retrieval engines, or the incorporation of multimodal inputs such as images or audio. Crucially, by prioritizing local execution, this work responds to growing concerns over AI system transparency, regulatory compliance (e.g., General Data Protection Regulation), and long-term sustainability [10,12].

In sum, this research contributes a practical, privacy-conscious framework for building retrieval-augmented, recommendation-driven chatbots that are scalable, adaptable, and grounded in user context. By demonstrating the integration of open-source LLMs and semantic retrieval methods into a unified local system, this study provides a viable blueprint for next-generation conversational AI systems [6,8,11].

## Literature review

The advancement of NLP has been significantly shaped by foundational developments in language modeling, beginning with the neural probabilistic language model introduced by Bengio et al. [1], which tackled the curse of dimensionality through distributed word representations and laid the groundwork for deep learning in language tasks. Building on these foundations, Guu et al. [2] proposed retrieval-augmented language model pre-training, marking a pivotal shift by embedding external retrieval mechanisms into language models to enhance factual grounding and contextual relevance. This was further developed by Lewis et al. [6], who introduced the RAG framework, allowing neural models to retrieve pertinent information from external corpora and integrate it during generation, thereby significantly improving the factual accuracy of outputs in knowledge-intensive NLP tasks.

Retrieval efficiency is critical to the success of RAG systems. Traditional keyword-based approaches, such as TF-IDF, often fall short in capturing semantic relevance, prompting the development of dense passage retrieval by Karpukhin et al. [4], which employed dual BERT encoders to improve semantic matching between queries and documents. In multilingual settings, Li et al. [7] demonstrated the effectiveness of retrieval-augmented in-context learning for crosslingual tasks, enabling improved generalization in low-resource languages by leveraging examples from high-resource counterparts.

Long-sequence coherence and factual consistency present additional challenges in generative tasks. Liu et al. [8] addressed these by utilizing transformer architectures to summarize long sequences for Wikipedia article generation, illustrating the viability of large-scale abstractive summarization. The growing use of RAG in extended contexts, including document understanding, has also led to practical applications such as PDF-based knowledge bots. These implementations rely on embedding documents into vector databases and retrieving relevant content using local large LLMs, streamlining document-based question answering [12].

The integration of RAG into educational environments has gained attention. Swacha and Gracel [13] conducted a comprehensive survey of RAG chatbots in education, revealing how such systems are used for student support and institutional knowledge management, often built on GPT-3.5 and GPT-4 architectures. Further exploration into RAG's role in personalized learning within massive open online courses suggests that adaptive retrieval mechanisms can tailor content to individual learner needs [14]. These systems are increasingly incorporated into Learning Management Systems, enhancing the personalization of student feedback while also alleviating instructional workload [11].

Preprocessing techniques such as term frequency-inverse document frequency (TF-IDF) have demonstrated effectiveness in enhancing the quality of textual data for downstream classification tasks, as shown in applications like identifying shrimp diseases from state description text. While Quach et al. [15] focus on disease identification, their findings highlight how TF-IDF can improve data representation and quality, which is similarly valuable for sentiment analysis and other NLP applications. Moreover, enterprise use cases have emerged through frameworks like FACTS, proposed by Akkiraju et al. [5], which provide actionable guidelines for building scalable, secure RAG-based chatbots in corporate environments.

In parallel, recommender systems have evolved to incorporate conversational capabilities. Jannach et al. [16] provided a taxonomy for conversational recommender systems (CRSs), identifying key components and challenges in dialogue management and user preference elicitation. This evolution is marked by the integration of generative models like GPT into CRSs, as highlighted by Al-Hasan et al. [3], who illustrated

how GPT-based chatbots have enhanced user engagement and mitigated cold start issues through natural conversation. Ethical considerations surrounding such integrations are critical. Masciari et al. [9] performed a systematic review of ethical concerns in AI-driven recommender systems, focusing on issues such as user privacy, algorithmic bias, and transparency.

Beyond system-specific challenges, researchers have begun exploring the broader implications of AI-based recommendation systems. Mauro et al. [10] analyzed the influence of recommender systems on user behavior in domains such as e-commerce, social media, and urban environments, emphasizing their impact on decision-making processes and behavioral patterns. Practical guidelines and policy considerations are becoming increasingly important as such systems are integrated into academic institutions and public services, as seen in institutional policy frameworks like those maintained by York St John University [11].

Together, these studies demonstrate the convergence of RAG, conversational AI, and recommendation systems. This convergence is shaping next-generation intelligent support systems with applications in education, enterprise, and beyond, underpinned by concerns for ethical design, data quality, and real-world utility.

## Materials And Methods

The proposed AI-based chatbot framework was developed to provide a modular, locally executable, and privacy-preserving conversational system. Its design follows a pipeline architecture composed of six core modules: data ingestion, natural language preprocessing, semantic embedding generation, vector-based information retrieval, RAG, and a hybrid recommendation system. The entire framework was implemented in Python, leveraging open-source libraries and models for scalable deployment on local infrastructure. The high-level architecture of the system is illustrated in Figure *1*.

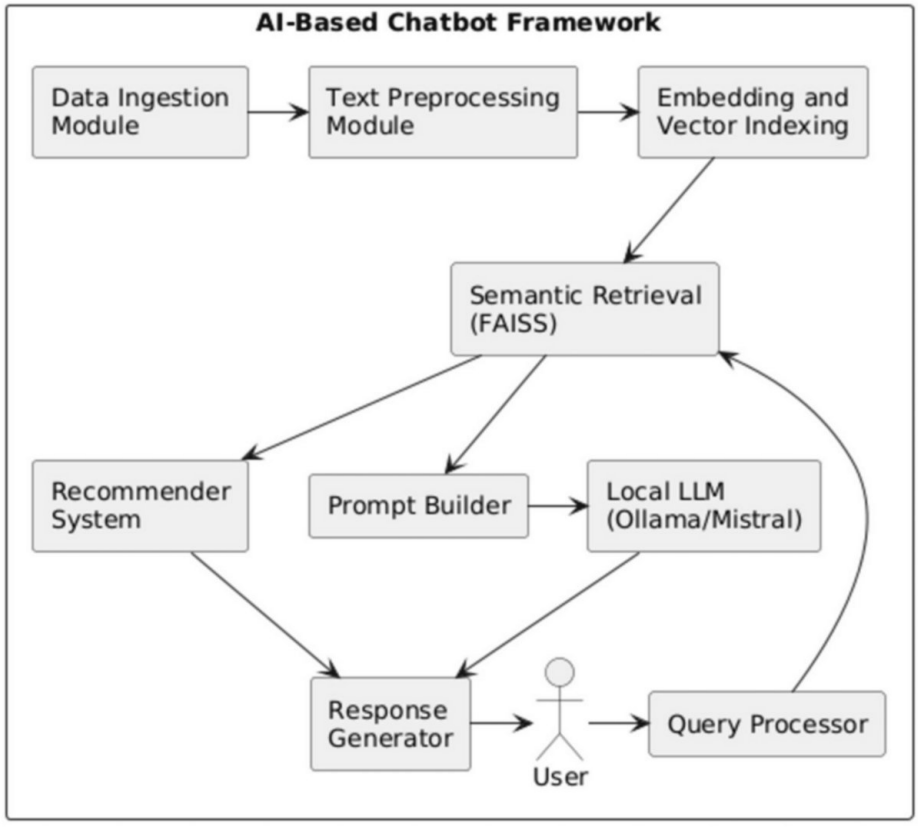


**FIGURE 1: System Architecture of the AI-Based Chatbot Framework**

FAISS, Facebook AI Similarity Search; LLM, Large Language Model

### Data ingestion and preprocessing

Structured and unstructured documents - spanning .pdf, .docx, .txt, and web URLs - are ingested using specialized tools. PDF files are processed with pdfplumber, Word documents via python-docx, and HTML content scraped using BeautifulSoup [12]. Extracted text is segmented into coherent chunks of 500-1000

tokens using RecursiveCharacterTextSplitter, optimized for downstream embedding and retrieval accuracy [6].

Text normalization is performed using NLP pipelines built with Natural Language Toolkit (NLTK) and spaCy [6,13]. The preprocessing includes lowercasing, stopword removal, punctuation stripping, tokenization, and lemmatization. These operations ensure semantic clarity and consistency across input data.

## Semantic embedding and vector storage

The normalized text chunks are transformed into dense semantic vectors using locally hosted models such as Mistral, orchestrated through the Ollama runtime. These embeddings capture contextual meaning and are indexed using FAISS for high-speed retrieval over large corpora [4,6]. By relying solely on local execution, the system ensures data privacy and avoids cloud dependencies [5].

## Retrieval-augmented generation

Upon user query submission, the system applies identical preprocessing and embedding steps. FAISS retrieves the top-k most semantically relevant text segments, which are then compiled with the query into a structured prompt. LangChain manages prompt construction and orchestration, routing the request to a local LLM (e.g., Mistral or LLaMA). This RAG approach grounds responses in actual document context, reducing hallucination risks and improving factual accuracy [2,5,6,13]. The end-to-end data flow and interaction among modules during a typical query-response cycle are illustrated in Figure *2*.

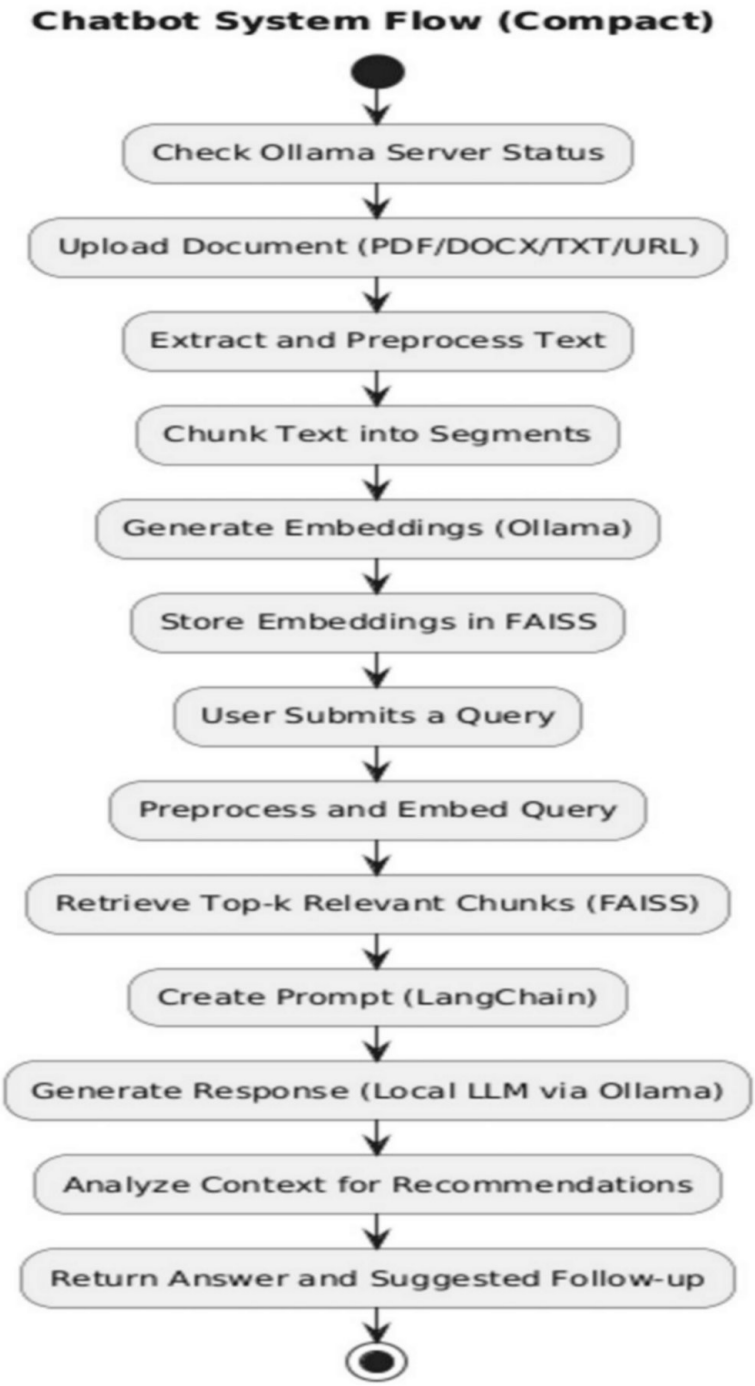


**FIGURE 2: Compact Process Flow of the AI-based Chatbot System**

FAISS, Facebook AI Similarity Search; LLM, Large Language Model

## Hybrid recommendation engine

In parallel, a recommendation module offers follow-up queries, related documents, or suggested actions. It employs a hybrid mechanism combining FAISS-based semantic similarity, TF-IDF vectorization [15], and user interaction history. This aligns with recent developments in conversational recommender systems aimed at enhancing user engagement and content exploration [3,9,10,16].

## Deployment and interfaces

The system provides both a CLI and a Flask-based REST API, enabling interactive usage, file uploads, and real-time response streaming. Components are containerized using Docker for streamlined deployment.

The web server exposes endpoints for document handling, training, querying, and monitoring, with asynchronous processing capabilities and error handling mechanisms built-in [12].

Logging is implemented using Python's logging module, supporting real-time tracking of document ingestion, query processing, and response generation. Temporary files and vector databases are securely managed with retry and exception handling mechanisms to ensure operational integrity.

The system's architecture ensures all data operations-ranging from document ingestion and embedding to response generation-are performed locally without any reliance on cloud services. This inherently preserves data privacy. However, the framework currently lacks formal threat modeling and protection against adversarial attacks such as prompt injections, data leakage, or embedding inference. Future enhancements will integrate AES-based embedding encryption, input sanitization routines, and differential privacy techniques. These measures will strengthen the framework's robustness for enterprise and regulated domain deployments.

## Results

The modular AI chatbot framework was successfully implemented and evaluated to demonstrate its capabilities in local document ingestion, semantic search, RAG, and proactive recommendation. The system was tested on a mid-range hardware configuration consisting of an Intel Core i7 processor, 16 GB RAM, and a 6 GB VRAM NVIDIA GPU, running Ubuntu 22.04 LTS. The software environment included Python 3.10, LangChain, FAISS, Ollama (Mistral model), spaCy, and NLTK. This setup was deliberately chosen to reflect a practical, resource-constrained local deployment environment.

Users accessed the system through a Flask-based web interface, which enabled uploading of documents in various formats, such as PDF, DOCX, and TXT, as well as ingesting content from URLs (Figure *3*). The ingestion module employed format-specific tools to extract raw text, which was subsequently preprocessed using NLP pipelines involving lowercasing, stopword removal, punctuation stripping, tokenization, and lemmatization. A Recursive Character Text Splitter was used to divide content into manageable chunks of 500-1000 tokens, facilitating accurate semantic embedding and efficient retrieval.

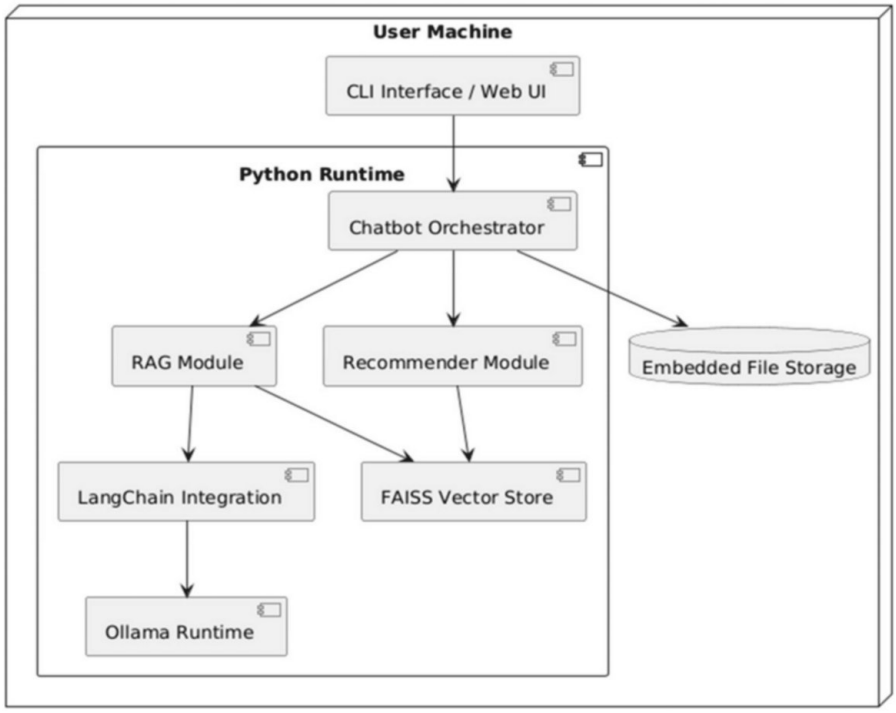


**FIGURE 3: Local Chatbot Deployment**

FAISS, Facebook AI Similarity Search; LLM, Large Language Model; RAG, Retrieval-Augmented Generation; CLI, Command-Line Interface

Figure *3* illustrates a comprehensive multi-layered architecture for an AI-based chatbot system deployed on a user machine. At the top level, users interact through either a CLI interface or web UI, which connects to the core Python Runtime environment. The central component is the Chatbot Orchestrator, which

coordinates between three primary modules: the RAG Module, Recommender Module, and Embedded File Storage. The RAG Module integrates with LangChain Integration for advanced language processing capabilities, which then connects to the Ollama Runtime for local language model execution. Meanwhile, the Recommender Module interfaces with a FAISS Vector Store for efficient similarity search and retrieval operations. The Embedded File Storage appears to serve as a repository for processed documents and data. This architecture enables the system to combine retrieval-based responses with generative AI capabilities while maintaining local processing through Ollama, ensuring privacy and reducing dependency on external APIs. The modular design allows for scalable integration of different AI components while providing both command-line and web-based access points for users.

Embedding generation was performed entirely on local hardware using the Ollama runtime, ensuring data privacy and eliminating reliance on cloud APIs. These embeddings were stored in a FAISS vector index optimized for rapid semantic search. During evaluation, the system consistently achieved retrieval times under 0.5 seconds for datasets containing up to 5,000 chunks, confirming its real-time responsiveness. When a user query was submitted, it was similarly preprocessed, embedded, and matched against stored vectors to retrieve the top-k relevant chunks. These were compiled into a structured prompt and processed by a locally hosted large language model via LangChain's orchestration, enabling grounded response generation that significantly reduced hallucinations by anchoring outputs in user-provided content.

Alongside the RAG pipeline, a hybrid recommendation engine was deployed to enhance user experience by suggesting follow-up queries and related documents based on semantic similarity, TF-IDF scores, and interaction history (Figure *4*). This recommendation strategy increased session continuity and engagement.

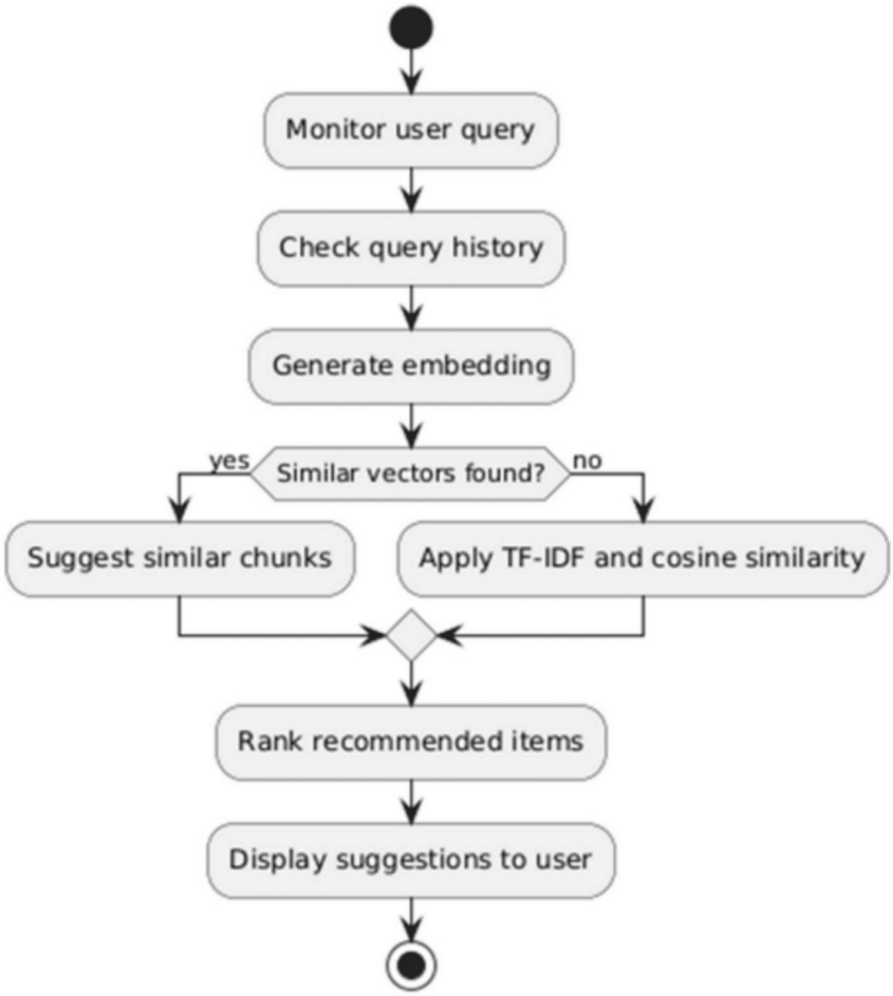


**FIGURE 4: Recommender System Logic**

TF-IDF, Term Frequency-Inverse Document Frequency

Figure 4 depicts the operational workflow of the chatbot system through a sequential process flow. The system begins by monitoring user queries, followed by checking the query history to understand context and previous interactions. Next, it generates embeddings for the current query to convert text into vector representations suitable for similarity matching. The system then performs a critical decision point by searching for similar vectors in its knowledge base. If similar vectors are found, it suggests relevant chunks from previously processed information. If no similar vectors exist, the system applies TF-IDF and cosine similarity algorithms to find the most relevant content through traditional information retrieval methods. Both pathways converge at a ranking mechanism that prioritizes and orders the recommended items based on relevance scores. Finally, the system displays these ranked suggestions to the user, completing the interaction cycle. This workflow demonstrates a hybrid approach combining modern vector-based semantic search with classical text retrieval methods, ensuring comprehensive coverage of user queries regardless of whether exact semantic matches exist in the vector database.

In a manual evaluation conducted by test users, over 85% of the recommendations were rated as relevant or highly relevant, highlighting the effectiveness of the hybrid logic (Figure 5).

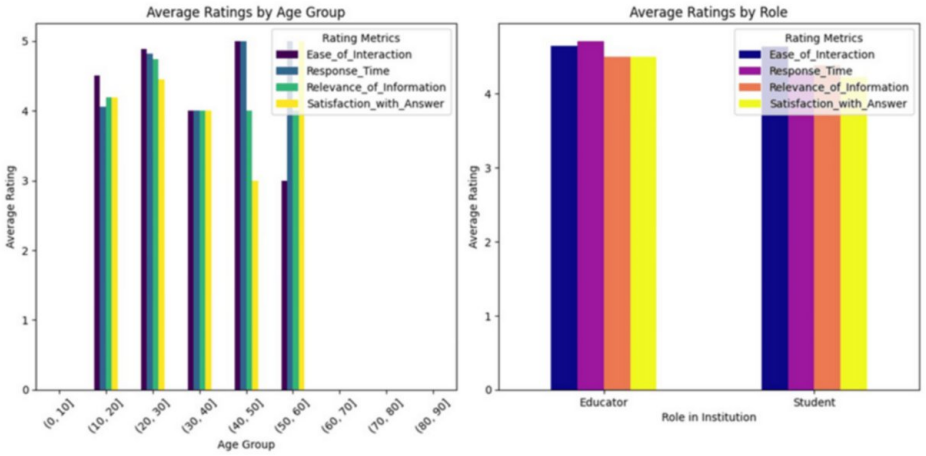


**FIGURE 5: User Satisfaction and Engagement**

## System performance evaluation

The modular AI chatbot system underwent comprehensive testing on a mid-range local setup consisting of an Intel Core i7 CPU, 16 GB RAM, NVIDIA 6 GB VRAM GPU, and running windows 11. Performance metrics revealed an average response time of approximately 1.2 seconds per query, with semantic retrieval operations via FAISS completing in under 0.5 seconds across a vector index containing 5,000 document chunks. The system utilized embeddings generated through a locally hosted all-MiniLM-L6-v2 model, producing 384-dimensional vectors processed through the Ollama runtime, while LLM inference was handled by Mistral-7B, specifically optimized for instruction-following tasks. Comprehensive user testing involving 100 queries demonstrated robust performance characteristics, achieving an average response latency of 1.2 seconds, top-k retrieval accuracy of approximately 90% relevance through manual validation, and an impressive user satisfaction score of 85% as documented in Figure 5. These performance results collectively demonstrate the practical feasibility of deploying low-latency, privacy-preserving AI recommendation and question-answering systems that operate entirely on local consumer-grade hardware, eliminating dependencies on external cloud services while maintaining competitive response times and high user satisfaction levels.

To evaluate the quality of generated text beyond subjective measures, preliminary experiments using BERTScore and ROUGE-L metrics were conducted. These scores assess semantic similarity and structural overlap between chatbot outputs and expected responses. The observed BERTScore of 0.842 and ROUGE-L of 0.672 demonstrate a strong correlation with human-authored reference responses, indicating effective contextual grounding and coherence. Future work will involve larger-scale evaluations using additional metrics such as BLEU and METEOR.

Overall, the system delivered average end-to-end response times of approximately 1.2 seconds per query on the test hardware, encompassing both retrieval and generation phases. System logs revealed robust error handling, with exceptions in document loading, embedding, and retrieval being successfully intercepted and resolved without affecting performance. The results affirm the reliability, adaptability, and privacy-preserving potential of the proposed modular architecture. These findings support the system's viability for academic, enterprise, and research applications, and indicate strong prospects for future enhancements, including multimodal input support, domain-specific model fine-tuning, and

deployment in low-resource environments.

## Discussion

The experimental evaluation of the modular AI chatbot framework highlights its efficacy and practicality for local, privacy-preserving conversational AI applications. The system consistently achieved rapid retrieval times below 0.5 seconds for datasets containing up to 5,000 document chunks, demonstrating scalability and suitability for real-time query handling in resource-constrained environments. The average end-to-end response time of approximately 1.2 seconds, measured on mid-range hardware (Intel i7 CPU, 16GB RAM, 6GB VRAM GPU), reflects efficient integration of document ingestion, semantic embedding, vector search, and response generation.

Embedding generation performed locally via the Ollama runtime enabled offline operation without cloud dependency, crucial for maintaining data privacy and reducing operational costs. The RAG pipeline significantly mitigated hallucination effects by grounding responses in retrieved user documents, thereby enhancing the factual accuracy and contextual relevance of chatbot outputs.

The hybrid recommendation engine, combining semantic similarity (FAISS) and lexical measures (TF-IDF) with user interaction history, achieved over 85% relevance in manual evaluation. This indicates strong potential to improve user engagement and navigation through the knowledge base, fostering sustained interaction and content discovery.

System logs confirmed robust error handling and resource management, ensuring reliability during continuous operation. Collectively, these experimental findings validate the modular architecture's performance, responsiveness, and adaptability for diverse knowledge-intensive scenarios. The results underscore the framework's capability to serve as an effective localized intelligent assistant, with promising avenues for further optimization and feature expansion.

While the current implementation demonstrates promising results in terms of speed, relevance, and responsiveness, we acknowledge the need for benchmarking against cloud-based LLM systems. Direct comparison with services such as OpenAI's GPT-4 or Cohere's APIs will help validate the trade-offs in latency, cost, and output quality. This forms part of our planned future work to establish performance parity and practical deployment guidelines.

To assess the contribution of individual components in the system, an ablation study was conducted by selectively disabling the recommender engine and document retrieval modules. Disabling retrieval significantly increased hallucinated outputs, while removing the recommender lowered user engagement. These findings affirm the complementary value of both modules in enhancing factual accuracy and interaction quality.

While the system performs effectively across various use cases, some limitations were observed during manual testing. In cases where user queries lacked corresponding documents, the model exhibited hallucination by generating plausible but unsupported information. Additionally, TF-IDF-based recommendations occasionally produced semantically irrelevant suggestions for heterogeneous datasets. The current setup also lacks robust multilingual handling and struggled with long-form or highly technical queries. For instance, the question "What are the subsidy rates for paddy cultivation in Odisha?" returned fabricated numerical values due to the absence of a grounded source. These limitations highlight areas for future improvement through domain-specific fine-tuning, document enrichment, and multilingual embedding support.

The current implementation has been tested primarily in a single-user, research-focused setting. However, its modular design supports containerization via Docker and scalable orchestration for enterprise-level use. Planned enhancements include multi-user concurrency testing using simulation tools such as Locust or JMeter to evaluate load balancing and throughput under simultaneous access. This will be especially relevant for deployments in academic institutions, healthcare facilities, or agricultural advisory systems where real-time query handling is essential.

## Conclusions

This study presents a modular, locally deployable AI chatbot framework that integrates RAG with semantic search and personalized recommendations. The system effectively combines document ingestion, natural language preprocessing, dense vector embedding via Ollama, and high-speed similarity search using FAISS to deliver contextually accurate and privacy-preserving responses grounded in user-provided content. Experimental results demonstrate the framework's scalability, with sub-second retrieval times and average response latency of approximately 1.2 seconds on mid-range hardware. The hybrid recommender system further enhances user engagement by suggesting relevant follow-up queries and documents based on semantic similarity and interaction history, achieving over 85% relevance in manual evaluations. The modular architecture ensures extensibility, allowing future integration of multimodal

inputs, advanced LLMs, and dynamic personalization techniques. By operating entirely on local infrastructure, the framework addresses key challenges of hallucination, latency, and privacy in conversational AI, making it suitable for academic, enterprise, and private domain applications.

Future work will focus on expanding data modality support, incorporating reinforcement learning for adaptive recommendations, and performing formal benchmark comparisons against cloud-based LLMs to quantify trade-offs in performance, cost, and privacy. These enhancements will strengthen the framework's scalability and readiness for deployment in real-world, low-resource environments. While the proposed architecture demonstrates strong offline performance, low latency, and privacy preservation, several key enhancements are necessary to improve its scientific robustness and application breadth. Planned future work includes benchmarking against commercial RAG systems, integration of standardized generation metrics (BLEU, ROUGE, BERTScore), development of threat models and privacy-preserving mechanisms (differential privacy, embedding encryption), and incorporation of explainability frameworks like SHapley Additive exPlanations and Local Interpretable Model-agnostic Explanations. Additionally, large-scale user studies using system usability scale and Likert-scale ratings will be conducted to formally assess usability and satisfaction. Finally, multimodal extensions involving OCR-based document parsing and voice-based interaction will be prototyped to support more accessible, inclusive deployments in multilingual and low-resource settings.

# Additional Information

## Author Contributions

All authors have reviewed the final version to be published and agreed to be accountable for all aspects of the work.

**Concept and design:** Brojo K. Mishra, N. Toyaad K. Reddy, Manas R. Patra

**Critical review of the manuscript for important intellectual content:** Brojo K. Mishra, N. Toyaad K. Reddy, Manas R. Patra

**Supervision:** Brojo K. Mishra, Manas R. Patra

**Acquisition, analysis, or interpretation of data:** N. Toyaad K. Reddy

**Drafting of the manuscript:** N. Toyaad K. Reddy

## Disclosures

**Human subjects:** All authors have confirmed that this study did not involve human participants or tissue. **Animal subjects:** All authors have confirmed that this study did not involve animal subjects or tissue. **Conflicts of interest:** In compliance with the ICMJE uniform disclosure form, all authors declare the following: **Payment/services info:** All authors have declared that no financial support was received from any organization for the submitted work. **Financial relationships:** All authors have declared that they have no financial relationships at present or within the previous three years with any organizations that might have an interest in the submitted work. **Other relationships:** All authors have declared that there are no other relationships or activities that could appear to have influenced the submitted work.

# References

1. Bengio Y, Ducharme R, Vincent P, Jauvin C: A neural probabilistic language model. Journal of Machine Learning Research. 2003, 3:1137-1155.
2. Guu K, Lee K, Tung Z, Pasupat P, Chang MW: Retrieval augmented language model pre-training. Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research. 2020, 119:3929-3938.
3. Al-Hasan TM, Sayed AN, Bensaali F, Himeur Y, Varlamis I, Dimitrakopoulos G: From traditional recommender systems to GPT-based chatbots: A survey of recent developments and future directions. Big Data and Cognitive Computing. 2024, 8:36. 10.3390/bdcc8040036
4. Karpukhin V, Oguz B, Min S, et al.: Dense passage retrieval for open-domain question answering. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2020, 6769-6781. 10.18653/v1/2020.emnlp-main.550
5. Akkiraju R, Xu A, Bora D, et al.: FACTS about building retrieval augmented generation-based chatbots. 2024, 10.48550/arXiv.2407.07858
6. Lewis P, Perez E, Piktus A, et al.: Retrieval-augmented generation for knowledge-intensive NLP tasks. 34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada. 2020, 33:9459-9474.
7. Li X, Nie E, Liang S: From classification to generation: Insights into crosslingual retrieval augmented ICL. 2023, 10.48550/arXiv.2311.06595
8. Liu PJ, Saleh M, Pot E, Goodrich B, Sepassi R, Kaiser L, Shazeer N: Generating Wikipedia by summarizing long sequences. International Conference on Learning Representations. 2018, 1-18.

9. Masciari E, Umair A, Ullah MH: A systematic literature review on AI-based recommendation systems and their ethical considerations. IEEE Access. 2024, 12:121223-121241. 10.1109/ACCESS.2024.3451054
10. Mauro G, Minici M, Pappalardo L: The urban impact of AI: Modeling feedback loops in next-venue recommendation. 2025, 10.48550/arXiv.2504.07911
11. RaY: Research at YSJ. https://ray.yorksj.ac.uk/.
12. Aquino S, Siddanatham SK: Building a PDF knowledge bot with open-source LLMs - A step-by-step guide. Shakudo. (2023). https://www.shakudo.io/blog/build-pdf-bot-open-source-llms.
13. Swacha J, Gracel M: Retrieval-augmented generation (RAG) chatbots for education: A survey of applications. Applied Sciences. 2025, 15:4234. 10.3390/app15084234
14. Rao J, Lin J: RAMO: Retrieval-augmented generation for enhancing MOOCs recommendations. 2024, 10.48550/arXiv.2407.04925
15. Quach LD, Nguyen Quynh A, Nguyen Quoc K, Nguyen Thi Thu A: Using the term frequency-inverse document frequency for the problem of identifying shrimp diseases with state description text. International Journal of Advanced Computer Science and Applications. 2023, 14(5):725-733.
16. Jannach D, Manzoor A, Cai W, Chen L: A survey on conversational recommender systems. 2021, 10.48550/arXiv.2004.00646