RAG System Development with Pydantic AI ChromaDB & Groq

Prof. Priyanka P. Kakade
Khemeshwarl Shelwante
Krushna Bhagwat Dhakne
Shreya Subhas Darade

Department Of Computer Engineering, Brahma Valley College of Engineering and Research Institute, Nashik, Maharashtra, India.

ABSTRACT

This project presents a Retrieval-Augmented Generation (RAG) based AI chatbot designed to provide intelligent and context-aware responses by combining retrieval and generation techniques. The system leverages Pedantic AI for structured data modelling and workflow orchestration, ChromaDB for vector-based document storage and retrieval, and Grog's Llama 3.3-70B model as the core language model for natural language understanding and generation. By integrating these technologies, the chatbot can access a large knowledge base, retrieve relevant information, and generate accurate and human-like answers. This approach enhances response reliability and enables efficient deployment for educational, enterprise, and research applications.

INTRODUCTION

1. Background of the Study

Foundation in Large Language Models (LLMs): Start by establishing that Large Language Models (LLMs) like GPT-4 or LLama revolutionized text generation but inherently faced limitations. Key LLM Limitations: Knowledge Cutoff: LLMs rely on static training data, making them unable to access or generate current, real-time information.
Hallucinations: LLMs can generate plausible-sounding but factually incorrect or fabricated information.
Lack of Specificity: They struggle to provide answers grounded in private, domain-specific, or proprietary knowledge (e.g., a company's internal documents). 1
The Rise of RAG: Introduce Retrieval-Augmented Generation (RAG) as the technical framework developed to address these core limitations by bridging the gap between static LLM knowledge and dynamic, external data sources.

2. Overview of RAG Systems

RAG as a hybrid AI technique that enhances the capabilities of LLMs by combining them with an **Information Retrieval (IR)** component.
Mechanism (How it works): Describe the fundamental two-step process:
Retrieval: When a user submits a query, the system first retrieves the most relevant documents, passages, or data chunks from an external knowledge base (often a vector database).
Augmentation and Generation: The retrieved information is then appended to the original user query, creating an **augmented prompt** that is fed to the LLM. The LLM then uses this specific, up-to-date context to generate a more accurate, grounded, and relevant response.

Key Components: Briefly mention the core elements: The **External Knowledge Base** (e.g.,

proprietary documents, web data), the **Embeddings/Vector Database** (for semantic search), and the **Large Language Model (LLM)** (the generator).

Importance of AI-Powered Retrieval Systems

Enhanced Accuracy and Trust: The most critical benefit. RAG grounds LLM responses in verifiable, factual external data, significantly reducing "hallucinations" and increasing user trust, especially since many RAG systems can **cite their sources**.

Access to Current and Domain-Specific Information: Allows AI applications to utilize real-time data (e.g., news, stock prices) or proprietary, internal documents (e.g., company policies, medical records) without the massive cost and time of retraining the entire LLM.

Cost Efficiency and Scalability: It is significantly more **cost-effective** and **scalable** to update an external knowledge base than to fine-tune or retrain a multi-billion-parameter LLM every time new data is introduced.

Improved Contextual Relevance: AI-powered retrieval, especially with technique like semantic search (using embeddings), ensures that the system finds information relevant to the *meaning* and *intent* of the query, not just exact keywords.

APPLICATIONS Of RAG SYSTEMS

Customer Support and Chatbots: Creating highly accurate conversational agents that can answer complex, domain-specific customer inquiries by pulling information from internal knowledge bases (e.g., product manuals, FAQs, past support tickets).

Knowledge Management and Enterprise Search: Enabling employees to quickly search and retrieve specific, summarized insights from vast volumes of internal documents (e.g., Bell's internal policies chatbot).

Healthcare and Legal Research: Aiding professionals by retrieving and summarizing the latest research papers, clinical guidelines, medical records (e.g., IBM Watson Health), or legal statutes and case files with high precision.

Content Creation and Summarization: Generating articles, reports, or concise summaries of long documents by reliably pulling in and synthesizing information from multiple sources.

Decision Support Systems: Providing financial analysts, managers, and R&D teams with real-time, consolidated insights from market trends, competitor reports, and internal data to facilitate informed decision-making.

PROPOSED SYSTEM

The proposed system integrates a front-end built with HTML, CSS, and JavaScript for dynamic and interactive user interfaces. The backend is powered by Node.js and Express.js, while CROMADB is used to manage user data.

OBJECTIVES

Primary Objectives

The primary objectives of a study or project on RAG systems are directly aimed at mitigating the core limitations of Large Language Models (LLMs) and achieving a **verifiably grounded**

response.

Mitigate Hallucination and Improve Factual Accuracy: To significantly **reduce the generation of false or fabricated information** (hallucinations) by grounding the LLM's response generation in specific, retrieved, external knowledge.

Integrate Real-Time and Proprietary Knowledge: To enable LLMs to effectively access and utilize **up-to-date, dynamic, or private domain-specific data** that was not included in their original static training sets.

Enhance Source Traceability (Explainability): To provide users with **clear, verifiable citations or references** to the source documents used to formulate the generated response, thereby increasing **transparency and user trust**.

Secondary Objectives

Secondary objectives focus on optimizing the RAG pipeline itself and extending its utility beyond basic factual accuracy.

Optimize Retrieval and Indexing Efficiency: To develop or evaluate advanced strategies (e.g., better chunking, hybrid search, re-ranking) that **maximize the precision and recall** of the retrieval mechanism, ensuring the *most* relevant context is passed to the LLM.

Improve Contextual Relevance: To ensure the generated output is not only accurate but also **highly relevant and coherent** within the specific context of the user's complex or multi-turn query.

Reduce Computational Cost and Effort: To achieve improved LLM performance and knowledge updates with greater **cost-efficiency** by relying on updating the external knowledge base (retrieval component) instead of requiring expensive and time-consuming LLM retraining or fine-tuning.

Explore Multi-Modal RAG: To extend the RAG framework to handle and retrieve information from **diverse data types** beyond text, such as images, audio, or structured data

The successful implementation of RAG technology yields significant benefits across technical, financial, and user experience domains.

LITERATURE SURVEY

Existing RAG Systems

Retrieval-Augmented Generation (RAG) refers to AI systems that **retrieve** relevant external information and then **generate** responses informed by that retrieval, rather than relying solely on a model’s internal knowledge. (McKinse&Company)

Key take-aways:

RAG consists of ingestion/indexing of knowledge, retrieval of relevant material for a query, augmentation of the query with that material, and then generation. (Pinecone)

Benefits: better accuracy, up-to-date responses, domain-specific knowledge without full retraining. (IBM)

Challenges: retrieval latency, data-quality dependence, risk of hallucinations or incorrect retrieval, complexity of the pipeline. (Proofpoint)

Recent reviews map out the taxonomy of RAG systems, including hybrid retrieval generation, multi-modal, reasoning enhanced retrieval, etc. (arXiv)

In short: RAG is a mature and growing paradigm for combining large-language models with external knowledge sources, but it still has operational and research challenges.

Pydantic AI Overview

Pydantic AI is a framework built on top of the popular Pydantic library (in Python) to help build AI agents and workflows, with structured output, type safety, and tool integration. (py.ai)

It supports multiple model providers and integrates with protocols like the Model Context Protocol (MCP) for tool-use and agent orchestration. (ai.pydantic.dev)

Strengths: clean API, production readiness, structured output enforcement. (Thoughtworks)

Limitations: While strong for structured output and agent building, managing large, complex multi-agent workflows (or advanced retrieval workflows) may require additional engineering. (From practitioner forums) (Reddit) Thus, Pydantic AI is a modern agent/AI orchestration framework that works well in structured workflow use-cases and is complementary to retrieval-based systems.

ChromaDB Overview

ChromaDB (also simply "Chroma") is an open-source vector database tuned for large language model applications: embedding storage, vector search, metadata filtering, etc. (Wikipedia)

It supports storing embeddings + metadata, efficient similarity search, and has been benchmarked for large-scale vector workloads. (Airbyte)

Use-cases: Documents + embeddings ingestion → vector store → retrieval for RAG pipelines or semantic search. (GeeksforGeeks)

Considerations: For high scale, architecture matters (write throughput, query latency, index maintenance) and retrieval relevance remains a challenge.

In short: ChromaDB is a key infrastructure component for retrieval pipelines (especially in RAG systems) to store and query embedding-based representations.

Groq Hardware Overview

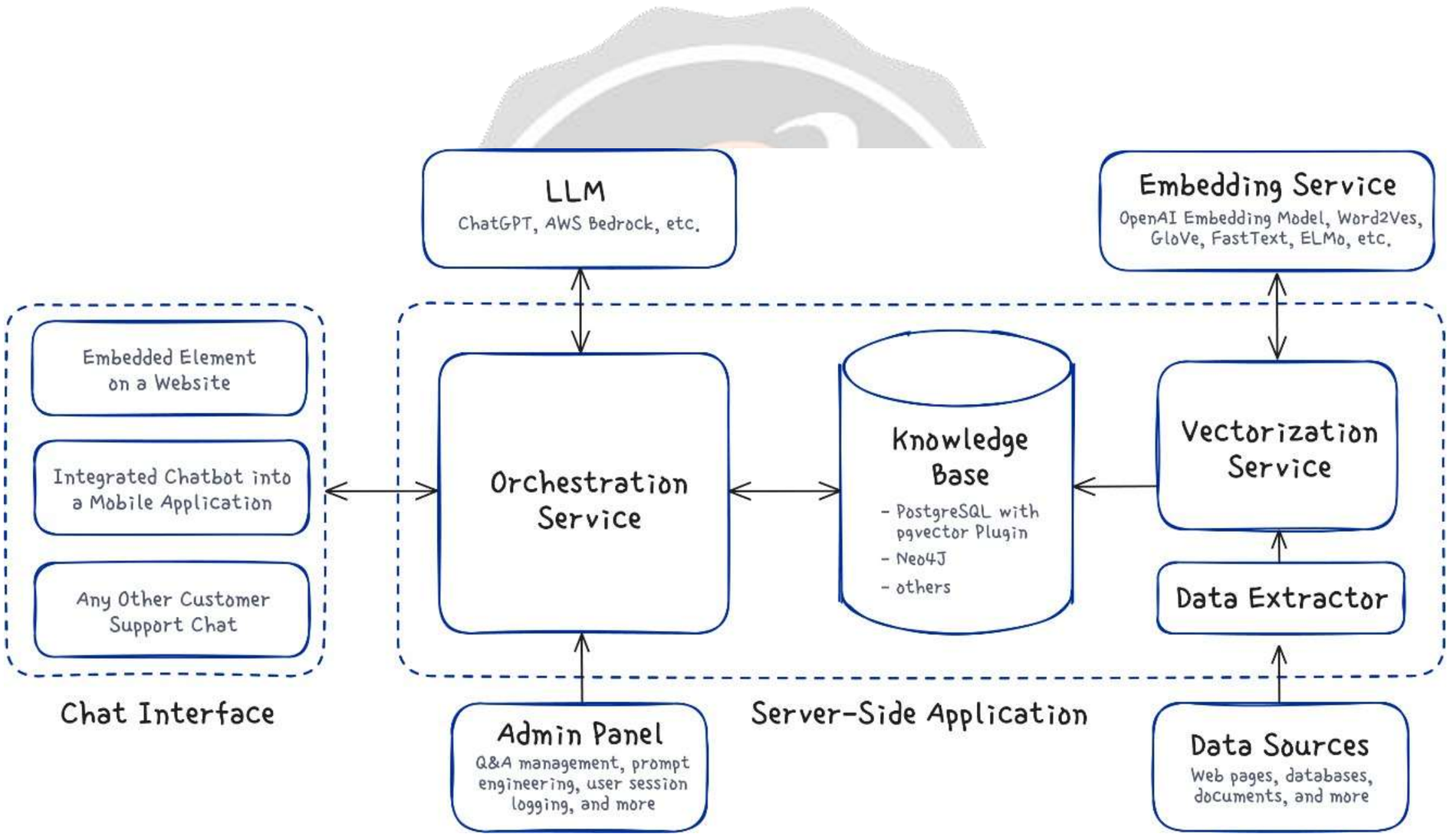
Groq builds custom inference hardware (the ‘‘LPU’’ – Language Processing Unit) designed specifically for fast and efficient inference of AI models (rather than training). (Groq)

Architectural features: co-located compute + memory, deterministic architecture, low latency, designed for production inference workloads. (docs.alcf.anl.gov)

Benefit in RAG/agent systems: When you’re retrieving info and running generative models in real time, inference latency matters; Groq’s hardware aims to cut that.

Limitations: As with any specialized hardware – adoption, ecosystem maturity, integration cost, and whether the models you use are optimized for it.In short: Groq provides a high-performance inference hardware layer that can support demanding real-time AI/agent deployments (such as retrieval + generation pipelines).

SYSTEM ARCHITECTURE



RESULTS AND DISCUSSION

Functional RAG System

A working Retrieval-Augmented Generation (RAG) system that can retrieve relevant documents and

generate accurate responses to user queries.

Efficient Knowledge Retrieval

Implementation of a **ChromaDB vector database** for fast and accurate semantic search.
Demonstration of embedding-based similarity measures (cosine similarity, dot product).

AI-Powered Response Generation

Use of **Pydantic AI agents** to generate structured, context-aware answers.
Capability to handle multi-turn queries and maintain conversational context.

Optimised Performance

Integration with **Groq hardware** for low-latency inference and high-throughput processing.
Benchmarking of system response time and accuracy.

User Interface / API

Simple and intuitive **frontend or API** for users to submit queries and receive responses.
Interactive demo showcasing retrieval and generation pipeline.

Documentation and Reports

Complete **project report** detailing system architecture, modules, data flow, feasibility, and results.
Technical documentation for installation, deployment, and usage.

Potential for Expansion

Modular design allowing integration of **additional datasets, embedding models, or AI agents** in future.
Foundation for advanced features like multi-modal retrieval or reasoning-enhanced generation.

CONCLUSION

The RAG System Development using Pydantic AI, ChromaDB, and Groq successfully demonstrates an integrated approach for knowledge retrieval and generation. By combining a vector-based retrieval system (ChromaDB) with a generative AI agent (Pydantic AI), the system can efficiently provide accurate, context-aware responses to user queries.Key achievements of the project include:

- Efficient retrieval of relevant documents using embeddings and similarity measures.
- High-quality response generation with structured and contextually appropriate answers.
- Optimized performance through the use of Groq hardware, ensuring low-latency inference.
- Modular and scalable design, enabling future enhancements such as multi-modal inputs or expanded knowledge bases.

This project highlights the practical applications of RAG systems in domains requiring fast, accurate access to large datasets. It serves as a foundation for further research and development in AI-powered knowledge systems and demonstrates the potential of combining modern AI frameworks, vector databases, and specialized hardware for building intelligent solutions.

Future Scope: The system can be extended to include real-time updates, multi-language support, and integration with cloud platforms to handle larger datasets and more complex queries.

ACKNOWLEDGEMENT

We would like to express my sincere gratitude to our supervisor, Prof. Priyanka P. Kakade, for their continuous support and guidance throughout this project. I would also like to thank my colleagues for their valuable feedback and assistance during the implementation phase.

REFERENCES

1. Lewis, M., et al. *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*. NeurIPS, 2020. https://arxiv.org/abs/2005.11401
2. Pydantic AI Documentation. *Pydantic AI Overview and Agent Framework*. https://ai.pydantic.dev
3. ChromaDB Documentation. *Vector Database for AI and Semantic Search*. https://www.trychroma.com
4. Groq Inc. *Groq Language Processing Unit (LPU) Overview*. https://groq.com
5. Pinecone. *Introduction to Retrieval-Augmented Generation (RAG)*. https://www.pinecone.io/learn/retrieval-augmented-generation/
6. IBM. *AI and Knowledge Retrieval Systems*. https://www.ibm.com/think/topics/retrieval-augmented-generation
7. Vaswani, A., et al. *Attention Is All You Need*. NeurIPS, 2017. https://arxiv.org/abs/1706.03762
8. Hugging Face. *Transfarmers and Embedding Models for NLP*. https://huggingface.co/docs/transformers
9. GitHub – Pydantic AI Examples. *Open-source RAG Implementation Examples*. https://github.com/pydantic-ai