

A PRESENTATION FOR PROJECT PROPOSAL

Submitted by

SURAJ RAMESH BABU-2513205

Under the guidance of

Dr. Seena Joseph

At



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Module Name: Research Methods and Professional Development
Module Code : ACFA7004

**Title of the Project: Development of an LLM-Driven
Autonomous Question Answering Agent with Retrieval
Augmented Generation (RAG)**

Abstract:

These days, both large and small businesses use AI agents. We have AI agents for a wide range of jobs, including handling customer service, assessing video footage, filling out forms at hospitals, and verifying legal papers. Businesses frequently spend hundreds of thousands of dollars on hiring customer service representatives who can comprehend a client's wants and address them in accordance with the company's policies. These days, having an intelligent chatbot to respond to frequently asked questions can effectively enhance customer support. This post will teach us how to use agentic RAG (Retrieval Augmented Generation), LangGraph, and ChromaDB to create a FAQ chatbot that can answer consumer questions in a matter of seconds.

Recent developments in Large Language Models (LLMs) have greatly improved machines' capacity to understand and produce text that is similar to that of humans. However, typical LLMs do not have access to real-time or domain-specific information; instead, they rely entirely on pre-trained knowledge.

Retrieval-Augmented Generation (RAG) combines external knowledge sources with LLMs to generate exact, context-aware, and current replies in order to get over this restriction.

The goal of this project is to create an autonomous question-answering agent that uses generative AI and semantic document retrieval to provide knowledgeable and trustworthy responses.

Interduction:

AI systems that are reliable, understandable, and accurate are essential in enterprise settings. Although simple RAG pipelines offer a basis for information retrieval, they often fall short in handling the subtleties of intricate, multi-departmental user requests. The change from "Naive RAG" to "Agentic RAG" signifies a move toward System 2 thinking, which is deliberate, multi-step reasoning that enables tool selection and dynamic routing. In order to minimize off-topic generation and increase user satisfaction through sentiment-aware escalation, this project employs an agentic framework that is intended to manage user interactions by dynamically directing inquiries to specialized retrieval paths.

Proposed Methodology:

The implementation uses a state-driven, modular architecture that is controlled by LangGraph. The process is divided into discrete functional nodes:

- **Query Classification Node:** Makes use of an LLM to map inquiries to predetermined organizational departments while conducting combined sentiment analysis and intent classification.
- **Contextual Routing Engine:** Uses conditional edges to guide the execution flow according to the outcomes of classification. An escalation process is triggered by negative emotion or unclear questions, avoiding automated generation to guarantee human review.
- **Vector-Based Retrieval Node:** Ensures that the recovered context is pertinent to the particular domain determined during classification by enforcing stringent department-level bounds using ChromaDB with metadata filtering.
- **Synthesis Node:** reduces the possibility of hallucinations by producing final responses that are firmly rooted in the recovered context.

Research Gaps & Further Directions:

Despite the operational efficacy of agentic orchestration, several gaps remain that warrant further investigation:

- **Interpretability and Explainability:** While agentic systems improve performance, it is still unclear how internal decisions are made, such as why a particular department was chosen. To map decision routes for compliance and auditing, "Explainable Agentic RAG" research is required.
- **Retrieval-Generation Alignment:** When an LLM uses retrieved context instead of parametric memory pre-trained knowledge, there is a permanent knowledge gap. For high-stakes situations, creating systems to gauge and enforce retrieval is essential.
- **Accuracy and Latency Trade-off:** The overhead introduced by multi-node graph traversal and iterative reasoning increases response

latency. Research is currently underway to optimize this orchestration without compromising the deliberative advantages of agentic processes.

- **Adversarial Robustness:** As these systems become more complex, they become susceptible to prompt injection and adversarial attacks aimed at bypassing department-level constraints.

Research Aim & Objectives:

Research Aim: To develop an agentic RAG workflow that improves response accuracy and human escalation procedures by classifying user intent and sentiment through stateful, graph-based orchestration.

- 1) **Design:** Using LangGraph to create a Directed Acyclic Graph (DAG) that separates nodes for generation, retrieval, and categorization.
- 2) **Assess:** Compare the efficacy of ChromaDB's department-based metadata filtering against a worldwide search baseline.
- 3) **Validate:** Analyse how well the system can use sentiment detection to activate human escalation nodes in comparison to a non-agentic baseline.

Proposed System:

The system consists of the following components:

- **Document Processing Module,** Collects domain-specific data, Performs text chunking and preprocessing.
- **Embedding Model** Converts text into high-dimensional vector representations.
- **Vector Database** Stores embeddings for fast semantic search.
- **Retriever** Fetches the most relevant documents based on user query.
- **LLM-Based Generator** Generates contextual answers using retrieved knowledge.

- Autonomous Agent Layer Manages workflow Decides when to retrieve, generate, and respond.

Expected Outcomes:

A precise and context-aware mechanism for responding to questions LLM replies with fewer hallucinations Semantic search that is quick and scalable AI-driven autonomous communication Reusable structure across several domains.

Applications:

- Enterprise knowledge assistants
- Academic support systems
- Healthcare information retrieval
- Customer support automation
- FAQ intelligent bots

Requirements:

Software requirements: Python / Google Colab

Tools to be used: LangGraph, Lang Chain, Chroma DataBase, OpenAI, Python-dotenv

Conclusion:

The suggested system combines generative AI with semantic retrieval to improve conventional methods of answering questions. For real-world intelligent knowledge retrieval applications, the autonomous agent-based RAG framework guarantees enhanced accuracy, contextual understanding, and scalability.

References:

1. K. I. Roumeliotis, R. Sapkota, M. Karkee and N. D. Tselikas, "Agentic AI With Orchestrator-Agent Trust: A Modular Visual Classification Framework With Trust-Aware Orchestration and RAG-Based Reasoning," in *IEEE Access*, vol. 14, pp. 26965-26982, 2026, doi: 10.1109/ACCESS.2026.3662282.
2. E. Marian Pasca, D. Delinschi, R. Erdei and O. Matei, "LLM-Driven, Self-Improving Framework for Security Test Automation: Leveraging Karate DSL for Augmented API Resilience," in *IEEE Access*, vol. 13, pp. 56861-56886, 2025, doi: 10.1109/ACCESS.2025.3554960.
3. S. Aboukadri, A. Ouaddah, A. Mezrioui and I. El Asri, "Leveraging RAG and LLMs for Access Control Policy Extraction From User Stories in Agile Software Development," in *IEEE Access*, vol. 13, pp. 116462-116472, 2025, doi: 10.1109/ACCESS.2025.3586203.
4. **Survey on Agentic RAG Architectures:** An analysis of the evolution from static to agentic paradigms.
5. **Vector-Graph Hybrid Models:** Studies exploring the integration of knowledge graphs and vector databases to reduce hallucination rates.
6. **Explainability in RAG:** A review of the necessity for transparent decision-making in multi-agent orchestration.