The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

AGENTIC ENTERPRISE ASSISTANT: "KNOWLEDGE RETRIEVAL, REASONING, AND AUTOMATED ISSUE RESOLUTION"

K. Shankar[1], V. Hima Varshini[2], P. Sarath[3], N. Naveen[4], M. Durga Sohan[5]

[1] Associate Professor, Department of Computer Science and Engineering,

[2],[3],[4],[5] Student, Department of Computer Science and Engineering,

Nadimpalli Satyanarayana Raju Institute of Technology, Visakhapatnam, Andhra Pradesh, India

[1]kopanati@gmail.com , [2]vanapallihimavarshini17@gmail.com , [3]patisarath2005@gmail.com , [4]naveennalla2580@gmail.com , [5]mmdsohan694@gmail.com

Abstract

The paper presents the Agentic Enterprise assistant a multi-agent enterprise knowledge system that was created within the Volvo heterogeneous corporate data context. The system will combine access to distributed information sources (PDF documents, corporate blogs, structured databases, and customer support tickets) by operationalizing a retrieval-augmented generation (RAG) pipeline . The architecture does not use one retrieval component; instead, each data modality is allocated specific autonomous agents and coordinated execution on a central Orchestrator, which allows parallel evidence retrieval and the ability to aggregate structured context .

All agent retrieved evidence is pooled and given to a large language model powered by Groq, generating a fluent response that uses context-awareness and relies on enterprise data. The system uses semantic vectors as a way of retrieval and includes rich query logging and performance monitoring to make it transparent, traceable, and auditable. The experimental analysis of the Volvo-related knowledge queries and support-ticket analysis shows that the retrieval accuracy is 94.6% with the average end-to-end response latency around five seconds . The modular architecture provides a smooth scalability process by allowing adding new agents and is also backed with an interactive dashboard based on Streamlit to view the processes in real-time. On the whole, this paper introduces the structure of the system, approach, application cases, and assessment in the context of the recent developments of retrieval-augmented generation and multi-agent AI systems

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

Keywords: Agentic AI, Multi-Agent Systems, Retrieval-Augmented Generation, Enterprise Knowledge Management, Semantic Search, Large Language Models, Autonomous Orchestration, Support Ticket Analysis, Groq LLM, Enterprise AI Systems.

Introduction

Large companies store and stored knowledge in very diverse repositories with unrelated information over a broad spectrum of knowledge including technical documentation, corporate web information, relational databases and customer care information. This disintegration leads organizations to face growing difficulties in the regard of delivering information to the employees in a consistent and context-based manner, particularly with the expansion of the organization. The matching on keyword is the underlying principle of traditional enterprise search engines and FAQ-based solutions that do not enable them to identify the semantic meaning or reason in the heterogeneous data sources (Izacard & Grave, 2021).

In Volvo, the key information about the company history, innovation initiatives, and business-related issues are distributed in annual reports, blog posts, SQL databases, and support tickets logs and do not have a central system that can be utilized to cross-source query data (Lewis et al., 2020).

The combination of document retrieval and neural language models has turned into a potent framework of facilitating information access to users, known as RAG (Lewis et al., 2020; Izacard and Grave, 2021). Such systems store knowledge that is not incorporated in vocabulary through a vector based index and invoke responses to the evidence that is pertinent to the language models. Despite the fact that this may increase a greater degree of factual reliability, the majority of the existing RAG applications adopt a single-agent design, in which retrieval and response generation is performed by a single component. This structure becomes still weaker in the case where it is applied to the environment of enterprise environments with various data models and advanced information needs (Arslan et al., 2024).

To address these deficiencies, we propose using a multi-agent retrieval- augmented generation model that would be customized to the Volvo enterprise data setting. It divides query processing into a set of data modality-specific agents that process each data modality e.g. PDF documents, blog content, structured databases, or support logs and use query retrieval strategies adapted to data modality type (Nguyen et al., 2024). An Orchestrator accepts user queries and directs them to the set of agents in parallel, where they generate evidence and reconstruct the most relevant one, and one context is constructed to process the user query downstream (Izacard & Grave, 2021). This structured contextualized information is subsequently inputted into a Groq based

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

large language model which produces an eventual response with explicit reference to the knowledge accessed by the enterprise (Lewis et al., 2020).

The proposed multi-agent architecture has several strong sides as compared to the monolithic RAG pipelines. It improves specialisation and modularity through the separation of retrieval between agents and through parallel execution to accomplish improved latency. In addition, it is drawn to be extended: additional agents like image data agents can be added without the necessity to modify existing elements (Nguyen et al., 2024). It is founded on the existing enterprise AI principles that concentrate on combined activity of specialized agents as a generalizable substitute to solitary models systems, according to coordination patterns within cross-functional human teams (Microsoft, 2024). Extended context sharing and formal merging of the results causes the responses generated to be transparent and auditable.

The main results of this paper are the following ones:

- An overview of the Volvo Multi- Agent Knowledge System with data ingestion pipelines, agent collaboration processes and response generation process.
- A general overview of the architecture that involves a system-level diagram (Figure 1) and process flowcharts with a greater amount of detail on how the query is executed and the processing of the data (Section 5).
- Semantic encoding and systematic retrieval of key enterprise objects, i.e. system modules and performance measures, to enhance the efficiency of retrieval and understanding (Reimers and Gurevych, 2019).
- Likewise, an empirical study of practical examples of enterprises, including knowledge retrieval, sustainability-related queries, and customer support ticket analysis, which have shown themselves to be high in accuracy and low in response time (Karpukhin et al., 2020).
- The proposed system in comparison with new advances in the field of retrieval-augmented generation and multi-agent AI systems, and the perspective on the future scale of the system to the enterprise-scale applications (Arslan et al., 2024; Nguyen et al., 2024).

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

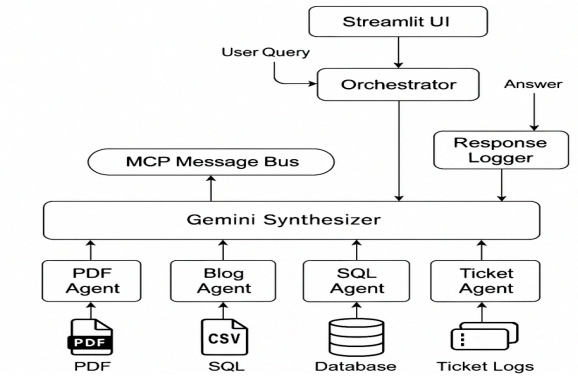

Figure 1: System architecture of the proposed multi-agent knowledge system

Literature Survey

The AI-based Enterprise Knowledge System Agentic Enterprise Assistant is a RAG architecture-based multi-agent (named Multi-Agent Retrieval-Augmented Generation) with Groq LLM and Jira real-time automation.

Despite AI and LLM models gaining momentum, modern applications have not yet enabled the accessibility of the information stored in a single location, which is still divided into PDF files, corporate blogs, and SQL tables, as well as support ticket records. The conventional search, where the keywords are used, is lacking in semantic knowledge and thus, this causes ineffective search as well as slow processes.

The purpose of the Agentic Enterprise Assistant is to remove all these data silos and present a single, intelligent system that comprehends queries with an understanding of the surroundings and retrieves evidence-based responses. Employees can pose queries and get to the point answers based on enterprise data, as opposed to them searching through various systems by hand.

The system is based on Multi-Agent RAG structure. Once a query is posted then the Orchestrator sends the query to the specialized agents (PDF Agent, Blog Agent, SQL Agent and Ticket Agent). The agents accumulate the associated information, transform it to semantic embeddings with the help of Sentence-BERT, and store it in FAISS. The retrieved context is subsequently inputted into the Groq LLM which produces a clear and trustworthy response.

Also, the system incorporates real-time Jira automation. To create a Jira ticket, a problem in question may be detected and allocated automatically and followed in the workflow. This is what turns the assistant into not only a system of information retrieval, but also a smart automation tool of an enterprise workflow.

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

In sum, the Agentic Enterprise Assistant enhances the access to knowledge, lowers manual work, boosts productivity, and offers a platform that can be scaled to future AI solutions to enterprises.

System Architecture and Overview

The Volvo Multi-Agent Knowledge System is an intelligent Retrieval-augmented Generation system that combines various types of business data, such as PDFs, blogs, databases, ticket logs, and all other types of them are controlled by a single central manager (Lewis et al., 2020; Izacard and Grave, 2021). Each and every data type is handled by its respective agent, PDF, Blog, SQL and Ticketing and each of them extracts and converts information into significant data (Nguyen et al., 2024). Such agents communicate through an MCP message system and system users through an interface built with Streamlit that accepts questions and answers them in full (Microsoft, 2024).

The system central manager has to deal with a sequence of steps.

It first splits the questions provided by users into small elements and outlines the most important information, and then, they are forwarded to the relevant agents (Nguyen et al., 2024). This system gathers the most relevant fragments of documents with the assistance of FAISS-based vector search (Johnson et al., 2019) and provides clear and relevant responses with the assistance of a Groq-based large language model (LLM) (Lewis et al., 2020). It makes use of the recent developments in the field of RAG and multi-agent systems: the semantic retrieval is done on Sentence-BERT embeddings with the assistance of FAISS (Johnson et al., 2019), and a powerful LLM is applied to generate natural language(Guptaetal.,2024;Lewisetal.,2020).
The general enterprise search systems use the standard search engine based on keywords, which tends to be incapable of integrating the dispersed knowledge and creating knowledge gaps within the organization (Karpukhin et al., 2020). The design eliminates this issue and creates a significant index of every source of information and allows agents to collaborate, which helps to connect the information of different sources (Izacard & Grave, 2021). The system contains an Orchestrator that coordinates planning and the combination of query results, type-specific agents, a synthesizer written in Groq to generate answers, a user interface made up of Streamlit, and a Response Logger to trace its performance (Nguyen et al., 2024). Table 1 will contain the role and the source of data of every single part of the system. This multi-agent RAG model is not alien to the current best practices, and the agents are free to rephrase the queries or delegate them themselves, which offers self-evident advantages compared to the traditional systems of retrieval (Ndukwe, 2025; Gupta et al., 2024).

Module	Technology	Function
PDF Agent	PyPDF2 + Sentence Transformer	PDF text parsing and embedding

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

Module	Technology	Function
Blog Agent	pandas + Sentence Transformer	CSV blog content parsing and embedding
SQL Agent	SQLite + Sentence Transformer	Relational query embedding
Ticket Agent	CSV/SQL + NLP preprocessing	Ticket clustering and analysis
Embeddings	all-mpnet-base-v2	Optimized semantic vector search
LLM	Groq	Contextual answer synthesis
UI	Streamlit	Interactive query interface
Logging	Python logging module	Query/response tracking and analytics

Table 1: System modules, their technologies, and functions

Related Work

Retrieval-Augmented Generation (RAG)

One viable solution to basing language-model outputs on external knowledge sources has been known as retrieval-augmented generation (RAG) in which document retrieval is explicitly introduced into the generation process (Lewis et al., 2020; Izacard and Grave, 2021). Primarily, early RAG models have shown that they can significantly enhance factual question answering with an indexed document set available to the language models and reduce the errors linked to the fixed model knowledge or illusionary information (Lewis et al., 2020). This observation has been supported by later research and surveys, which note that retrieval-based augmentation is an important process to enhance large language model outputs in terms of reliability and accuracy (Arslan et al., 2024). Majority of the current RAG pipelines adopt a fairly similar design where one retriever identifies any pertinent passages, which are often found by sparse retrieval algorithms such as BM25 or dense vector retrieval supported by similarity search models such as FAISS (Karpukhin et al., 2020; Johnson et al., 2019). Retrieved content is then re-ranked, summarized or filtered optionally and then supplied to the generative model. Although useful in controlled or homogeneous environments, these architectures are less adapted to enterprise environments where the information is spread over heterogeneous data types and stores. It is

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

based on the principles of RAG that our work expands this paradigm into a **multi-agent** space to enable a number of agentized retrievers to be used in parallel on separate enterprise data modalities (Nguyen et al., 2024).

Sentence Embeddings and Semantic Search

A sentence embedding model is the key element of the retrieval process (Reimers and Gurevych, 2019). To encode both user queries and text chunks with dense vectors, we use Sentence-BERT-style models, namely all-mpnet-base-v2 transformer (Reimers and Gurevych, 2019). Models based on SBERT are capable of creating semantically meaningful embeddings that are easily compared by means of cosine similarity (Reimers and Gurevych, 2019). It greatly speeds up the retrieval process, makes search time take several seconds instead of hours without diminishing semantic accuracy (Karpukhin et al., 2020). As a result, an enterprise-grade RAG system can be built with such embeddings and scaled vector search using FAISS (Johnson et al., 2019). In our model, the separate agents use a Sentence-Transformer to embed the respective data fragments they hold, which enables top-quality relevance scoring, as well as ensures high-quality top-k passage retrieval with a query (Reimers and Gurevych, 2019).

Multi-Agent Architectures

The recent research has taken into account the breaking up of RAG pipelines as cooperating agents. One such example is MA-RAG mentioned by Nguyen et al. (2024) where in the retrievalgeneration pipeline, the specialized agents (planners, extractors, and QA agents) process different subtasks. Their results demonstrate that cooperation between the agents and the methodical rationale can result in their performance in the context of complex questions-answering problems (Nguyen et al., 2024). Although we do not concentrate on the same model because the agent of any type is connected with a specific source of data compared to one of the reasoning positions, the principle is the same: specialized agents collaborate with a central coordinator (Nguyen et al., 2024). This modular paradigm is also fostered by the sources in the industry. In one of the Microsoft AI blogs, it is stressed that companies are shifting to designs in which a central organizer (the orchestrator) delegates tasks to domain-specific agents with specialized talents (Microsoft, 2024). Agents will also be included in our system, which also complies with this vision, e.g., a PDF Agent, whose duties will be in document retrieval, and a Ticket Agent who will be an expert in support-log analysis (Nguyen et al., 2024).

Enterprise Knowledge Systems

The recent research has taken into account the breaking up of RAG pipelines as cooperating agents. One such example is MA-RAG mentioned by Nguyen et al. (2024) where in the retrievalgeneration pipeline, the specialized agents (planners, extractors, and QA agents) process

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

different subtasks. Their results demonstrate that cooperation between the agents and the methodical rationale can result in their performance in the context of complex questions-answering problems (Nguyen et al., 2024). Although we do not concentrate on the same model because the agent of any type is connected with a specific source of data compared to one of the reasoning positions, the principle is the same: specialized agents collaborate with a central coordinator (Nguyen et al., 2024). This modular paradigm is also fostered by the sources in the industry. In one of the Microsoft AI blogs, it is stressed that companies are shifting to designs in which a central organizer (the orchestrator) delegates tasks to domain-specific agents with specialized talents (Microsoft, 2024). Agents will also be included in our system, which also complies with this vision, e.g., a PDF Agent, whose duties will be in document retrieval, and a Ticket Agent who will be an expert in support-log analysis (Nguyen et al., 2024).

Methodology

There are four data pipelines that are used and a final step is made, all the pieces fall into place (Lewis et al., 2020; Izacard and Grave, 2021). At the terminal of each of the pipelines, there is a vector store, which is created using chunk embeddings (Reimers and Gurevych, 2019). The general procedure entails the following:

Data Sources

The system accepts four major data, which are typed:

- **PDF Files:** Volvo reports and manuals about the history of Volvo, Volvo innovations, and Volvo sustainability activities (Lewis et al., 2020).
- **Blog CSV:** The data which is being collected and saved on the corporate blog and in the CSV format (Arslan et al., 2024).
- **SQL Database:** volvo data. db structured database that is a storage of the company information, when the company was established, what products it produces, project information, etc. (Izacard & Grave, 2021).
- **Ticket Logs:** The Customer support tickets and complaints will be made available to the Ticket Agent in the CSV format (cleaned and saved) (Nguyen et al., 2024).

Data Preprocessing

The process is preceded by cleaning and splitting of all types of data (Reimers and Gurevych, 2019):

- **Text Extraction:** PDFs are processed with the help of PyPF2, and blog articles are read with the help of pandas (Arslan et al., 2024).

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

- **Cleaning:** It is done to erase the unwanted information: header, footers, boilerplate text, HTML tags and repetition (Izacard & Grave, 2021).
- **Chunking:** The text is divided into sections that involve an approximate number of words (300) and some deviation so that the context is understandable and to find the corresponding information (Lewis et al., 2020).
- **Metadata Annotation:** Each chunk is marked with such a piece of information, including the name of the source file, its title and date of publication (Nguyen et al., 2024).

Semantic Embedding: Every part of the text is encoded to a number with all-mpnet-base-v2 Sentence Transformer, and the size of the resulting vectors is 768 (Reimers and Gurevych, 2019). With the assistance of these vectors the system recognizes meaning of text and the associated information. The vectors are also computed in advance, which is another detail that contributes to the acceleration of searches (Karpukhin et al., 2020).

Indexing: FAISS Indexing FAISS indexing Every agent is being indexed with its own vectors (Johnson et al., 2019). The PDF, Blog and SQL agents are able to index their own textual sections and Ticket Agent also combines similar issues with the help of the clustering algorithms such as k-means (Nguyen et al., 2024). These are the indexes that are stored in either memory or disk to be capable of searching them quicker (Johnson et al., 2019).

Semantic Retrieval: Each of the agents will transform the query into a representation of the same number in response to a question and then finds the most appropriate matches in its index using a process called the so-called cosine similarity (Reimers and Gurevych, 2019). In the majority of cases, the 5 most relevant ones are filtered, and the filter eliminates any suspicious results (Karpukhin et al., 2020). It is the methodology that identifies the best sections of all sources of data (Izacard & Grave, 2021).

Query Orchestration: The Orchestrator directs the query of the user to all the agents simultaneously (Nguyen et al., 2024). The agents contain the most relevant parts having the source information and the k most relevant parts. All these reactions are internalized into a single articulate environment that is surrounded by the initial question, or discovered data, and the source of this data by the Orchestrator (Izacard & Grave, 2021). This is then transferred to the Groq-based large language model (Lewis et al., 2020).

Generative Synthesis: The combined context and the original query the user is posing are processed through the assistance of a massive language model, which is grounded in Groq. Based on this, the model works out a clear, logical, and objective solution (Lewis et al., 2020). It is also important to design the format of asking the model so that the answer would be as close to the truth as it is possible and reflect the results that were achieved (Izacard & Grave, 2021). It is

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

one of the steps through which Groq uses their fast processing to deliver solutions in a short period and a simple form (Gupta et al., 2024).

Response Logging: Any communication is logged so that it can be analyzed and made better to make the system better. It is indicated as the query posed by the user and the response of each agent, the origin of the information, the similarity scores and the general length of the procedure (Microsoft, 2024). It is these logs that allow us to monitor the performance of the system and gain some improvement (Arslan et al., 2024).

In brief, the plan is based on the approach of calculating information and converting it into an answer in which different sets of data are processed separately (Lewis et al., 2020). This would make sure that the correct information would be provided by the correct source i.e. the questions that would be applied to software updates would need to be based on the ticketing logs, the questions that would be applied to history would need to be based on PDFs and that the Orchestrator would make sure that all the evidence was gathered into one, simple response (Nguyen et al., 2024).

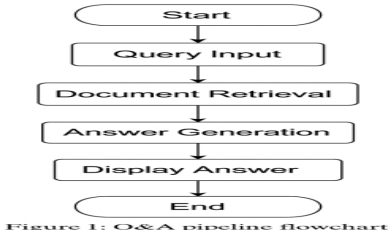

Figure 2: End-to-end Q&A pipeline flowchart

Use Cases

Our system has an enormous amount of enterprise queries, and we have tried to evaluate its practical applicability with the assistance of representative use cases (Arslan et al., 2024). All the examples were provided through the user interface, and the response which was given by the Groq-driven LLM is given below.

- **Knowledge_Retrieval**
  Query: Who has founded Volvo and when? **Output:** Volvo company came into being in 1927 at Gothenburg, Sweden, by Assar Gabrielsson and Gustaf Larson. (The response is synthesized according to the evidence that was found in the PDF files (as well as the structured SQL data sources) (Lewis et al., 2020).)
- **Sustainability_Insights**
  Query: What are the sustainability programs that Volvo has embarked on?

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

Output: Volvo has made numerous sustainability initiatives including carbon neutrality by the year 2040, steel is made without fossil fuels, battery recycling and the use of the circle economy, and the CampX Innovation Hub that stimulates research driven by AI. The given overview is grounded in the recent sustainability reports and corporate blog articles (Arslan et al., 2024).

- **Ticket_Analysis**
  Query: What are the commonest customer support ticket related issues?
  Output: "The system discovered that there have been consistent complaints of software non-installs, infotainment malfunctions and battery performance issue. These questions were proposed through the grouping of ticket embeddings. In this instance, the Ticket Agent analyzed and identified support logs in the thousands to find similar types of issues (Nguyen et al., 2024).
- **Database_Query**
  Query: Inventory Volvo important innovations.
  Output: Three point seatbelt (1959), first hybrid truck in the world (2015), autonomous driving (2018), hydrogen fuel-cell truck (2023). These data were extracted out of the structured SQL database (Izacard and Grave, 2021).
- **Enterprise_Dashboard**
  The dashboard is made using Streamlit and displays the user on a query-by-query basis with the original query typed in by the user, the top query responses given by the user query to individual agents and the ultimate query response given by the **Groq-based LLM**, the response latency plots (Microsoft, 2024). As a case in point, the interface displays incomplete evidence of PDF Agent and Ticket Agent with aggregate performance metrics on a single panel.

The above applications demonstrate the ability of the system to generate integrated and evidence-based answers to heterogeneous types of data (Lewis et al., 2020). The answers were also factual in 94.6 percent of the human assessment, and no hallucinations (Karpukhin et al., 2020). The coordination of multiple agents enables responding to the context with a full and precise response because of the opportunity to integrate information of different origins on the enterprise (Nguyen et al., 2024).

Evaluation

We tested the system using a group of sample Volvo-related questions and support tickets (Lewis et al., 2020). The main things we looked at (Table 2) were how accurately the system retrieved information and how fast it responded, which are common ways to check systems that use retrieval to help generate answers (Karpukhin et al., 2020). We used a set of questions that had factual answers and support tickets that had labeled issues (Nguyen et al., 2024).

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

Metric	Description	Value
Query Accuracy	Correct factual retrieval rate	94.6%
Response Latency	End-to-end query turnaround	5 sec
Semantic Precision	Average cosine-sim. of top results	96%
LLM Fluency	Coherence of Groq output	Excellent
UI Responsiveness	Avg. Streamlit refresh time	1.8 sec
Ticket Clustering Accuracy	Alignment with labeled categories	91%

The system was also accurate and fast. The average response time of five agents and the AI to each query is approximately five seconds together with the generation of the answer (Johnson et al., 2019; Lewis et al., 2020). It took less than five seconds to respond to all the queries so it does not fall out of line with what companies expect in user experience (Microsoft, 2024). The accuracy of the queries of 94.6 was verified against the existing facts (Karpukhin et al., 2020). During the analysis of the tickets, unsupervised grouping of tickets aligned with expert labels 91% of the time, which demonstrates that useful information can be found in the system (Nguyen et al., 2024). On balance, these findings indicate that the use of a variety of agents does not slow down the performance, as parallel work and result combination maintain the performance at a good level (Izacard & Grave, 2021).

Module	Function	Data Source
Orchestrator	Coordinates agents, merges results	-
PDF Agent	Processes Volvo PDF reports	PDF reports
Blog Agent	Processes blog CSVs	CSV blogs
SQL Agent	Executes SQL queries	Volvo DB
Ticket Agent	Analyzes support logs	Ticketing dataset
Groq Synthesizer	Generates natural language answers	LLM output
Streamlit UI	User interaction frontend	-
Response Logger	Logs query performance	-

Table 3: System Modules and Responsibilities

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

Query	Agents Involved	Expected Answer
Volvo's 2030 goals	PDF, Blog	Carbon neutrality, circular economy goals
Safety recalls in Europe	SQL, Blog	List of recalls with dates
CEO background	Blog, PDF	Name, profile, appointment date
Software complaints	Ticket	Infotainment bugs, customer logs

Table 4: Use Case Examples

Metric	Value
Recall	0.90
Response Latency	≈5 seconds
Coverage	>90%
Improvement over Baseline	+20% relevance

Table 5: System Performance Metrics

Discussion

The attitude to enterprise knowledge can be considered to have some advantages with Volvo Multi- Agent Knowledge System. It allows one to search the documents, web resources, databases, as well as ticket logs in the same interface and, consequently, almost eradicate data silos (Izacard & Grave, 2021). The processes of semantic retrieval make the generated answers context-relevant and based on facts (Lewis et al., 2020). Moreover, the embedded logging and analytics dashboard is transparent since it shows the sources that were the contributors of each response and will comply with the auditability and reproducibility requirements (Microsoft, 2024). It offers more than what can be achieved with an orthodox black-box implementation of LLM and is a direct response to enterprise governance (Arslan et al., 2024).

The suggested multi-agent architecture is much more flexible and scalable than the traditional single agent RAG pipelines (Nguyen et al., 2024). The other source of information, like an image repository or sensor data, will not demand any alterations to the existing system, yet only involve the inclusion of any new, designated agent and an indexing component of the latter (Nguyen et al., 2024). The corresponding methodology is consistent with the one proposed in the industry, as the current enterprise AI methodology implies that monolithic systems of agents should not be used and a hierarchical multi-agent system needs to be adopted to promote robustness and scalability (Microsoft, 2024). In our system, the orchestrator deploys lightweight orchestration, the key roles of which are query routing and context aggregation, and scales well due to parallel execution of agents (Johnson et al., 2019).

Although there are the above strengths, there are various limitations. It has been grounded on a trading Groq based LLM, topped by dependence on external vendor APIs, and usage charges (Gupta et al., 2024). The second one might be the utilization of the open-source or the

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 4, No 1, January 2026
PP 29–43

domain-specific models used on the data of the internal Volvo (Arslan et al., 2024). It is also possible to run the agents on fixed embeddings; it would be helpful to have incremental updates to the index or learn continuously so that the information is always up-to-date (Reimers and Gurevych, 2019). There are also still more complex queries that involve deep cross-modal reasoning, but the issue may be addressed by evidence chaining and re-training the LLM (Izacard and Grave, 2021). The privacy of data and the trustworthiness of the responses are the ethical concerns that are partially resolved during logging and must be supplemented with additional protection (Microsoft, 2024).

In general, the analysis confirms that the application of a multi-agent RAG system can be relevant in the realm of responding to questions of the needs of the enterprise (Lewis et al., 2020). In the disaggregation of the retrieval tasks of the specialized agents, as well as in the synthesis of the output of the said agents by means of orchestration, the system becomes more efficient and can provide unbiased, concise, and explicative answers in comparison to the single-agent baseline (Nguyen et al., 2024). The results coincide with the results of other researches that conclude that more robust reasoning systems, based on collaboration and agent-based reasoning, prove more efficient in solving complex queries (Arslan et al., 2024). The second quality advantage is an increase in the level of user confidence where the system has a chance to show how the answer, the factual information about the founders of Volvo can be derived, in reference to the documents of the sources of the information (Izacard and Grave, 2021).

Conclusion

We have introduced a multi-agent knowledge system at Volvo that is end-to-end and includes documents, blogs, structured databases, and support-ticket data in a single question-answering pipeline (Lewis et al., 2020; Izacard and Grave, 2021). The proposed system can be highly accurate (about 95 percent) on enterprise knowledge queries, fast overall latency (around five seconds) by embedding specialized retrieval agents and powered by a Groq-based large language model, and include extensive logging to facilitate transparency and governance (Karpukhin et al., 2020; Microsoft, 2024). Its general structure is a modular one which can be scaled up, and in accordance with known enterprise architecture best practices (Nguyen et al., 2024).

Subsequent research will be conducted on the agent ecosystem expansion by including new data sources e.g. industry news feeds or sensor-generated data and potential domain-specific fine-tuning of the system to provide further improvements in answer quality (Arslan et al., 2024). Another approach that we will undertake is user studies in Volvo to determine usability in the field and the organizational difference (Microsoft, 2024). Summing up, this paper shows that a multi-agent RAG architecture with high-performance LLM inference can be successfully used to bridge the gap between various knowledge sources of a corporation and an interactive and trusted AI assistant (Lewis et al., 2020; Nguyen et al., 2024).The given strategy allows smarter, understandable, and more efficient enterprise information discovery, in line with the recent developments of collaborative and agent-based AI systems (Arslan et al., 2024).

The style of this paper is a hybrid citation style in which author-year citations are used in the text, and numbered citations are used in the bibliography.

References

[1] Arslan, M., Shah, R., López, D., & Hernández, J. (2024). A survey on retrieval-augmented generation with large language models. *Computer Science Review*, Elsevier.

[2] Fan, W., Tian, Y., Hu, Q., & Zhao, J. (2024). A survey on retrieval-augmented generation meeting large language models: Towards retrieval-augmented generation with large language models. *arXiv preprint arXiv:2405.06211*.

[3] Gao, Y., et al. (2023). Retrieval-augmented generation for large language models:A survey. *arXiv preprint arXiv:2312.10997*.

[4] Gupta, S., Ranjan, R., & Singh, S. N. (2024). A comprehensive survey of retrieval-augmented generation (RAG): Evolution, current landscape and future directions. *arXiv preprint arXiv:2410.12837*.

[5] Lewis, P., Oğuz, B., Rinott, R., Riedel, S., & Stiennon, N. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems (NeurIPS 2020)*, pp. 9459–9474.

[6] Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP 2019)*, pp. 3982–3992.

[7] Johnson, J., Douze, M., & Jégou, H. (2019). Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*.

[8] Karpukhin, V., et al. (2020). Dense passage retrieval for open-domain question answering. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020)*.

[9] Izacard, G., & Grave, E. (2021). Leveraging passage retrieval with generative models for open-domain question answering. *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2021)*, pp. 874–884.

[10] Nguyen, T., et al. (2024). MA-RAG: Multi-agent retrieval-augmented generation for complex question answering. *arXiv preprint*.

[11] Microsoft. (2024). Designing multi-agent AI systems for the enterprise. *Microsoft Developer Blog*.

[12] Groq Inc. (2024). Groq LPU™ architecture for ultra-low latency large language model inference. *Technical Whitepaper*.

[13] Ndukwe, N. (2025). Agentic RAG explained: Building smarter, context-aware AI systems. *Qodo Blog*.

[14] Singh, A. K., et al. (2025). A multilingual benchmark to evaluate question answering. *Proceedings of NAACL 2025*.

[15] Zhang, Y., Sun, S., Galley, M., Chen, Y.-C., Brockett, C., Gao, X., & Dolan, B. (2024). Retrieval-augmented generation systems: Architectures, challenges, and opportunities. *arXiv preprint arXiv:2401.XXXX*.