Article

Not peer-reviewed version

Low Code RAG-LLM Framework for Context-Aware Querying in Electrical Standards, Design, and Research

Soham Ghosh * and Gaurav Mittal

Posted Date: 7 July 2025

doi: 10.20944/preprints202507.0537.v1

Keywords: low-code AI automation; retrieval-augmented generation (RAG); contextual RAG; vector database search; knowledge graphs; document-centric question answering; national electrical code (NEC); large language models (LLM)

Preprints.org is a free multidisciplinary platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This open access article is published under a Creative Commons CC BY 4.0 license, which permit the free download, distribution, and reuse, provided that the author and preprint are cited in any reuse.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025

doi:10.20944/preprints202507.0537.v1

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Article

Low Code RAG-LLM Framework for Context-Aware Querying in Electrical Standards, Design, and Research

Soham Ghosh [1,*] and Gaurav Mittal [2]

[1] Electrical Engineering Manager, Black & Veatch, Overland Park, Kansas 66211, USA
[2] Enterprise Solutions Architect, Black & Veatch, Overland Park, Kansas 66211, USA
* Correspondence: sghosh27@ieee.org.com

Abstract

Large Language Models (LLMs) have demonstrated remarkable capabilities in natural language understanding and generation. However, in domain-specific applications such as electrical engineering, standalone LLMs are often susceptible to hallucinations and inaccuracies, limiting their reliability for technical querying. Retrieval-Augmented Generation (RAG) frameworks address this limitation by grounding LLM outputs in external, verified sources, thereby improving factual accuracy. Despite these advantages, most RAG implementations demand significant programming knowledge and infrastructure setup, creating a high barrier to entry for practicing engineers and engineering researchers who may not have a software development background. To address this challenge, this manuscript introduces a low-code RAG-LLM framework built using the N8N automation platform, enabling users to construct and deploy advanced RAG pipelines with minimal technical overhead. The framework supports enhancements such as reranking, contextual retrieval, and knowledge graph-augmented generation, all of which are implemented in a modular and reproducible manner. Extensive testing was conducted to evaluate the impact of each enhancement on retrieval accuracy and generation quality, using documents from National Fire Protection Agency (NFPA), IEEE Xplore Digital Library, and CIGRE. The strengths and limitations of these approaches are discussed in detail to guide practical adoption. All associated N8N JSON workflow files are made available with this manuscript to facilitate replication and customization by engineering professionals and researchers.

Keywords: low-code AI automation; retrieval-augmented generation (RAG); contextual RAG; vector database search; knowledge graphs; document-centric question answering; national electrical code (NEC); large language models (LLM)

---

1. Introduction

Large Language Models (LLMs) have transformed various fields, including natural language processing and artificial intelligence. However, despite their remarkable linguistic capabilities, LLMs are notably susceptible to hallucinations, instances where they generate outputs that are factually incorrect or nonsensical without clear justification. This issue stems from multiple factors intrinsic to LLM design and operation, including the reliance on large, potentially unfiltered training datasets and the auto-regressive nature of these models, which can lead to inconsistencies in the generated content [1,2]. Hallucinations can manifest as fabricated facts, illogical conclusions, or even entirely inaccurate responses, posing significant risks in critical applications such as clinical decision-making and information retrieval [3]. For instance, studies have demonstrated that hallucination rates in various LLMs can range from 50% to 82% under different operational settings, underscoring the pervasiveness of this phenomenon in machine-generated text [4].

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025 doi:10.20944/preprints202507.0537.v1

Given the profound implications of hallucinations, particularly in sensitive domains such as professional disciplines such as engineering, healthcare, and law, there is a pressing demand for innovative approaches to enhance the reliability of LLM outputs. One promising strategy is Retrieval-Augmented Generation (RAG), which merges LLM capabilities with retrieval-based methods to improve accuracy and context relevance. RAG effectively allows LLMs to access up-to-date external knowledge sources during the text generation process, substantially mitigating the limitations posed by fixed training datasets [5,6]. In practical applications, the integration of RAG has been shown to markedly improve the performance of LLMs in critical situations, such as in fast identification of necessary sections within engineering design code and standard or in querying knowledge from a cluster of engineering documents. By enabling LLMs to draw upon supplementary data, RAG systems enhance coherence and factual correctness in the outputs, demonstrating their significant utility across various high-stakes scenarios [7].

Furthermore, recent advancements in understanding and managing hallucinations align with the growing utilization of RAG techniques. By relying on RAG mechanisms, which interleave generative capabilities with rigorous retrieval processes, LLMs can self-correct erroneous outputs by cross-referencing information dynamically sourced from trusted databases. This not only fosters a more reliable computational environment but also elevates user trust in automated systems [8]. As LLMs evolve, the incorporation of RAG methodologies appears to be a critical step toward ensuring that these technologies can produce accurate, context-aware, and trustworthy responses across an expanding array of applications. While Retrieval-Augmented Generation (RAG) frameworks offer significant advantages over standalone Large Language Models (LLMs) by grounding responses in authoritative sources, several practical challenges limit their broader implementation in domain-specific fields such as engineering. These barriers are:

1. The technical entry barrier remains high, since constructing a performant RAG-LLM pipeline typically requires substantial programming expertise, which many engineering professionals and researchers may lack.
2. To be truly effective, an RAG-LLM application must integrate seamlessly with the user's existing document repositories, such as Google Drive or SharePoint. However, implementing such integrations introduces additional layers of complexity, often involving authentication protocols, API configurations, and document parsing logic.
3. Achieving optimal performance from a RAG pipeline frequently necessitates incorporating advanced enhancements such as reranking strategies, contextual retrieval mechanisms, or knowledge graph-based augmentation. While these techniques improve retrieval precision and output relevance, they are often technically demanding and require familiarity with multiple NLP tools and frameworks. These nuances collectively restrict the accessibility and practical deployment of RAG-LLM systems for many practitioners in engineering and applied research settings.

The authors have directly encountered these barriers in real-world engineering research and application settings. To address the steep entry requirements and technical complexities, a low-code, low-entry-barrier framework has been developed using open-source LLMs and modular RAG pipelines that can be readily deployed through accessible platforms such as LM Studio, AnythingLLM, and N8N. This approach eliminates the need for extensive programming, enabling practitioners to focus on the engineering content rather than software infrastructure. Multiple workflow implementations have been demonstrated using the N8N automation environment, covering use cases such as document retrieval, reranking, contextual querying, and knowledge graph integration. Extensive performance testing has been conducted using engineering documents and technical standards drawn from sources such as the National Fire Protection Association (NFPA), Institute of Electrical and Electronics Engineers (IEEE), and the International Council on Large Electric Systems (CIGRE). The results confirm that a low-code RAG-LLM architecture can deliver both usability and high performance in technically demanding domains.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025 doi:10.20944/preprints202507.0537.v1

This manuscript makes methodological advancements at the intersection of low-code retrieval-augmented generation (RAG) systems and evaluation of technical document querying aligned for practicing power system professionals. The specific contributions are:

1. *Low-entry RAG-LLM pipeline implementation:* A practical illustration of a low-barrier RAG-LLM pipeline is provided, utilizing LM Studio and AnythingLLM to enable secure, local document retrieval and language model inference. The accompanying code for parsing and embedding domain-specific documents is made available and can be accessed through the 'Data Availability' section of the manuscript.
2. *Practitioner-oriented evaluation and design considerations:* The manuscript frames the design and evaluation of RAG workflows with a focus on practitioner guidance, particularly for electrical engineers and applied researchers working in low-code environments such as N8N. Special attention is given to the challenges encountered by traditional RAG systems when processing dense tabular data and multi-layered exceptions commonly found in technical codes and standards.
3. *Contextual retrieval workflow* via *N8N:* A contextual RAG workflow built in N8N is introduced, demonstrating improved handling of tabular structures and 'exception logic' through orchestrated document processing and dynamic retrieval logic. The complete workflow is made publicly available and can be accessed via the 'Data Availability' section to support reproducibility and practical adoption.
4. *Knowledge graph-based retrieval workflow* via *N8N and Infranodus:* A knowledge graph-augmented RAG workflow is presented using N8N in conjunction with Infranodus, enabling semantic structuring of engineering documents to uncover hidden relationships between technical concepts. This approach supports exploratory querying and cross-referencing across interconnected standards, particularly useful in complex design and compliance tasks. The complete workflow, including graph generation, integration, and retrieval logic, is made publicly available in the 'Data Availability' section to facilitate reproducibility and practical application.

The manuscript is organized as follows: section 2 introduces the fundamentals of Retrieval-Augmented Generation (RAG), outlining its core components and typical workflow architecture. Section 3 presents an easy-to-adopt, private RAG framework built with open-source LLMs, leveraging LM Studio and AnythingLLM for local deployment. Section 4 discusses low-code integration using the N8N platform for document retrieval, where prompts were extensively tested and ranked using a question set derived from the NFPA 70 National Electrical Code (NEC) 2017. This section also documents key insights into the scenarios where traditional RAG-LLM pipelines perform well and where they fall short. Section 5 addresses these limitations through targeted enhancements, focusing on reranking strategies and contextual RAG workflows. Comparative performance scores are presented for different rerankers (such as bge-reranker-base and Cohere Reranker v3.5) across multiple embedding models including OpenAI and Voyage. Section 6 provides an advanced implementation of a multi-brain, knowledge graph-based RAG system, highlighting its applicability for exploratory querying across interconnected engineering standards. Practical strategies for minimizing API usage and associated costs are also outlined. Finally, Section 7 concludes with a summary of findings, discusses current challenges, and identifies directions for future work.

2. Fundamentals of Retrieval-Augmented Generation (RAG)

Large Language Models (LLMs) such as GPT-4 and LLaMA exhibit impressive capabilities in generating coherent and contextually appropriate text. However, a well-documented limitation of these models is their tendency to "hallucinate"- i.e., produce plausible-sounding but factually incorrect or unverifiable information. This phenomenon arises because LLMs generate outputs based on statistical correlations learned from their training data rather than a grounded understanding of real-world facts or domain-specific knowledge. To mitigate this limitation, Retrieval-Augmented Generation (RAG) has emerged as a promising architectural paradigm. In RAG, the generation

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

process is coupled with a retrieval component that dynamically fetches relevant documents or context passages from an external knowledge corpus. This is conceptually akin to providing a model with access to a domain-specific textbook at inference time, enabling it to consult authoritative sources before constructing its responses.

By integrating retrieval into the generation loop, see Figure 1, a RAG system can significantly enhance factual accuracy and domain alignment. For instance, in specialized domains such as power systems engineering or regulatory compliance, where precise terminology and references are essential, RAG ensures that responses are anchored to source materials curated for that field. This makes RAG particularly well-suited for applications involving knowledge-intensive question answering, code documentation, engineering reasoning and queries, scenarios where hallucinations can have high stakes. However, RAG is not a substitute for domain adaptation or continual learning. It does not enable a model to learn new syntactic structures, programming languages, or stylistic conventions. Instead, it provides a targeted augmentation mechanism for improving response fidelity within predefined information boundaries. In this sense, RAG enhances retrieval rather than representation, making it more appropriate for precision-oriented tasks than for general-purpose domain training.

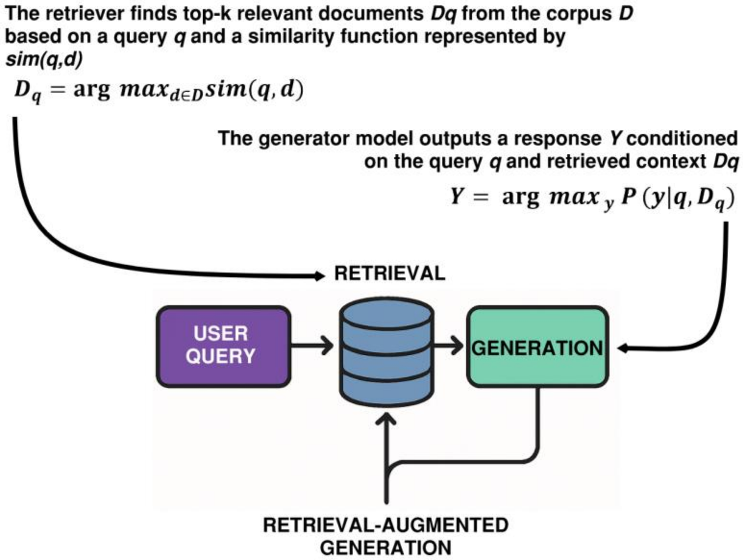

The retriever finds top-k relevant documents *Dq* from the corpus *D* based on a query *q* and a similarity function represented by *sim(q,d)*
$D_q = \arg\, max_{d \in D} sim(q, d)$
The generator model outputs a response *Y* conditioned on the query *q* and retrieved context *Dq*
$Y = \arg\, max_y\, P(y|q, D_q)$
RETRIEVAL
USER QUERY
GENERATION
RETRIEVAL-AUGMENTED GENERATION

Figure 1. Illustration of RAG architecture, where a user query initiates a retrieval process from a knowledge base (e.g., vector database or document store).

While hosted LLM interfaces such as ChatGPT allow users to upload domain-specific documents, such as an electrical engineering textbook or an IEEE standard, and perform retrieval-augmented querying, several critical limitations arise when relying on such cloud-based services for professional or industrial use. These are:

1. Data sovereignty and confidentiality, become a pressing concern. Uploaded documents leave the confines of personal or corporate workstations, potentially violating internal data handling policies, intellectual property agreements, or regulatory requirements, especially when proprietary designs, sensitive specifications, or restricted standards are involved.
2. Token and context length limitations inherent to hosted models often pose technical barriers. Large documents, such as multi-chapter standards or comprehensive handbooks, frequently exceed the maximum token window (even with chunking and summarization techniques), leading to truncated context or incomplete retrieval, which can compromise the accuracy of the generated response.
3. Hosted solutions typically lack persistent, user-controlled storage of parsed knowledge. Once the document is uploaded and used for a single session, the embedding and indexing layers are ephemeral and inaccessible to the user. This prevents engineers from building long-term,

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

reusable vector databases that can evolve with ongoing projects or organizational knowledge needs.

These limitations underscore the need for self-hosted RAG pipelines, systems where practitioners can download and operate LLMs locally, parse their own engineering documents, and store the resulting embeddings in a vector database of their choice (e.g., FAISS, Pinecone, or Qdrant). Such a setup empowers users with complete control over the retrieval layer, custom indexing strategies, and integration with in-house tools or platforms. For electrical engineers, this approach ensures that sensitive design documentation, grid planning standards, or equipment protocols can be queried securely, reliably, and repeatedly, without reliance on third-party APIs or external cloud services. It also opens the door to fine-tuned document access policies, auditability, and long-term knowledge retention, which are essential in high-assurance engineering environments.

3. Opensource LLM and Private RAG-LLM Pipeline for Engineering Professionals

Open-source large language models (LLMs) offer the distinct advantage of local deployability, allowing users to run inference securely within their own computing environments, without reliance on third-party APIs or external cloud services. This is particularly beneficial for retrieval-augmented generation (RAG) applications in engineering domains where data privacy, reproducibility, and system-level control are essential. To identify suitable models for local deployment, engineers can review the Hugging Face Open LLM Leaderboard [9]. By filtering for “mid-range” models (typically in the 14B to 32B parameter range), as shown in Table 1, users can select models that offer a balance between inference performance and hardware feasibility.

Table 1. Open LLM leaderboard filtered by ‘mid-range’ models and sorted by weighted average of normalized scores from all benchmarks (as of June 2025).

Model	Average score across all benchmarks	IFEva†	GPQA‡	CO_2 cost
Qwen2.5-test-32b-it	**47.37%**	78.89%	15.21%	29.54 kg
Horizon-AI-Avengers-V1-32B	47.34%	79.72%	14.99%	7.95 kg
FluentlyLM-Prinum	47.22%	80.90%	18.23%	21.25 kg
Qwen.5-14b-v1.0-e3	47.09%	73.24%	22.26%	**1.56 kg**
Qwen2.5-32B-Instruct-abliterated-v2	46.89%	83.34%	15.66%	13.49 kg

†Instruction-Following Evaluation (IFEval): This metric tests the model's ability to follow explicit formatting instructions. ‡Graduate-Level Google-Proof Q&A (GPQA): This metric scores the LLM performance based on PhD-level knowledge multiple choice questions in science. Boldface scores represent highest scores or lowest carbon footprint.

The private and locally executed RAG-LLM leverages two open source softwares, LM Studio and AnythingLLM; see Figure 2. Starting with LM Studio, open source LLM models of choice can be downloaded in quantized formats (such as GGUF) and launched with minimal configuration. The configurations that should be set are the system prompt, the randomness temperature, and the top-K sampling parameter. To implement a complete RAG pipeline, AnythingLLM, another open-source application, is used to create a local retrieval workspace. PDF documents such as IEEE standards, electrical codes, or equipment manuals should first be converted to a Markdown format (file extension .md) before being uploaded to AnythingLLM. Markdown works better for RAG because it provides clean, structured, and token-efficient text that enables accurate chunking and retrieval, unlike the noisy and layout-heavy content in PDFs.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

AnythingLLM further allows users to configure 'Text Chunk Size' and 'Text Chunk Overlap', critical parameters that govern the granularity and contextual continuity of document embeddings. AnythingLLM connects directly to LM Studio (which should run locally), where the LLM resides, and uses the locally hosted model to generate responses to queries based on the embedded content. This configuration ensures that both the model and the knowledge base remain entirely within the user's local system, maintaining full control over data, execution, and reproducibility without reliance on cloud-based inference or third-party services.

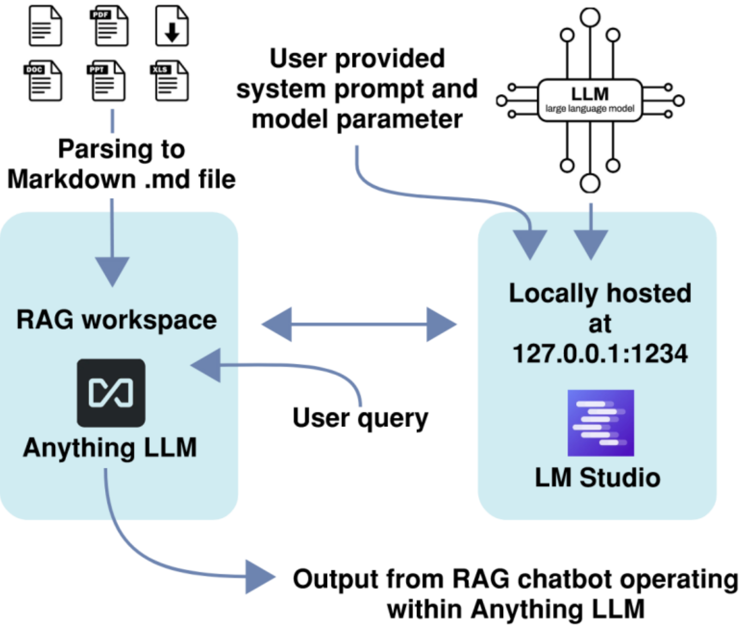

User provided system prompt and model parameter
LLM
large language model
Parsing to Markdown .md file
RAG workspace
Anything LLM
Locally hosted at 127.0.0.1:1234
LM Studio
User query
Output from RAG chatbot operating within Anything LLM

Figure 2. Illustration of a private and locally executed RAG-LLM pipeline using LM Studio and Anything LLM.

4. Low-Code Integration with N8N for Document Retrieval

While the combination of LM Studio and AnythingLLM offers a fast, secure, and locally deployable solution for building retrieval-augmented generation (RAG) systems, it remains somewhat limited in terms of workflow automation, integration breadth, and orchestration flexibility. For more sophisticated use cases, such as integrating document ingestion, vector storage, and multi-channel querying, an extensible low-code automation framework becomes necessary. This is where N8N, an open-source, node.js-based workflow automation tool, excels. N8N enables users to design automated pipelines that integrate AI capabilities with enterprise-grade business process automation. It supports local deployment, ensuring data privacy, and can be extended with custom logic, APIs, and plugins, making it well-suited for engineering, document intelligence, and RAG applications.

In the context of this manuscript's use case, a representative N8N-based RAG workflow is illustrated in Figure 3. The workflow functions as follows:

1. *Document upload:* Engineers or technical staff upload documents, such as PDF datasheets, word-based specifications, or standard manuals into a designated Google Drive folder.
2. *Automated ingestion:* N8N monitors the Google Drive folder for new uploads. When a file is detected, it triggers a processing pipeline that extracts the content, optionally parses it to Markdown or plain text, and embeds the text using a user-defined embedding model.
3. *Vector storage:* The resulting embeddings are stored in a local or cloud-hosted vector database (e.g., Qdrant, Chroma, or Pinecone) for later retrieval.
4. *Query interface:* End users submit queries via various channels, such as Gmail, Slack, or a web-based chatbot. The N8N agent retrieves the top-K relevant chunks from the vector database.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

5. *LLM generation:* These retrieved passages are then passed to an LLM of the user's choice, which is hosted locally and generates a grounded, context-aware response.

This modular and event-driven architecture not only decouples the ingestion and querying phases but also allows users to automate multi-step, cross-platform knowledge workflows, enabling a more scalable and maintainable RAG-LLM solution in engineering environments.

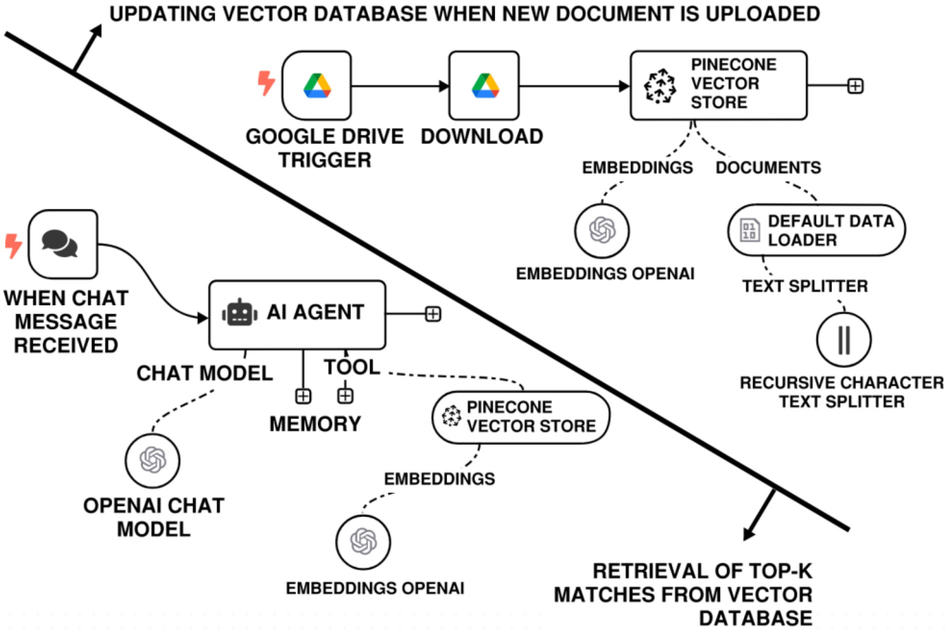


Figure 3. N8N RAG-LLM workflow with a google drive-based retrieval system offering a low entry barrier for users.

To validate the effectiveness of the proposed RAG-LLM pipeline, the system was tested using a curated set of domain-specific queries referencing the National Electrical Code (NEC), a complex and highly structured regulatory document widely used in electrical engineering practice. The NEC codebook (in parsed textual form) was uploaded by the user into the RAG system, enabling grounded document retrieval for inference. The evaluation queries were designed to test the system's ability to:

1. Locate and extract numerical values from NEC tables (e.g., minimum burial depths, conductor ampacity ratings, overcurrent protection limits).
2. Interpret codebook exceptions and conditional clauses, which often appear as footnotes or structured rule deviations.
3. Retrieve relevant sections and apply contextual logic (e.g., identifying requirements that vary by installation type, voltage class, or application environment).

Each generated response was evaluated against user-defined ground truth answers, see Table 2, simulating expert validation. For scoring, DeepEval's correctness metric with the GEval criteria was user, which assesses semantic alignment, factual grounding, and contextual completeness between the model's response and the reference answer.

The GEval metric offers a granular assessment based on the following:

1. *Grounding:* Did the model cite or synthesize relevant text from the retrieved NEC section?
2. *Exactness:* Did it accurately reproduce or infer key numeric thresholds and regulatory conditions?
3. *Verifiability:* Are the statements presented traceable to the original document sections?

This evaluation framework ensures that the RAG-LLM system not only generates plausible text, but also provides code-compliant, reference-anchored answers suitable for technical decision-making in real-world engineering workflows.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025 doi:10.20944/preprints202507.0537.v1

Table 2. Validation of traditional RAG-LLM responses using an N8N workflow on query tasks based on the 2017 National Electrical Code (NEC). Score is computed using DeepEval's correctness with criteria [GEval] metric.

Q. Prompt: I am trying to find the table in the NEC that deals with single phase AC motor and associated full load current. **A. Ground truth:** Table 430.248 of the NEC deals with single phase AC motor and associated full load current. **A. Actual output:** According to the National Electrical Code (NEC), the section dealing with single-phase AC motors and their associated full-load currents can be found in Article 430, which is titled "Motors, Motor Circuits, and Controllers." Specifically, you may refer to Table 430.248 of Article 430 for information on the minimum full-load current required for a single-phase AC motor based on the nameplate rating of the motor. **Score: 0.9072185128712421** **Reason:** The actual output correctly identifies Table 430.248 as the relevant section for single-phase AC motors and their full-load currents, aligning with the expected output. It provides additional context by mentioning Article 430 and its title, which enhances factual accuracy and comprehensiveness. There are no contradictions or omissions, but the extra detail slightly exceeds the expected output's simplicity.	**Q. Prompt:** What is the minimum cover required per Section 300.5 of the NEC, for direct buried cables running under a parking lot based on table 300.5. **A. Ground truth:** For direct buried cables running under a parking lot based on table 300.5, minimum cover required is 24 inches. **A. Actual output:** According to Table 300.5 of the NEC, the minimum cover required for direct-buried cable installations running under a parking lot is 24 inches (609 mm) based on the values provided in the table. This value applies to the top surface of the finished grade, and other factors may also be considered when determining the appropriate cover for your specific installation. **Score: 0.8592666599995407** **Reason:** The actual output correctly identifies the minimum cover required as 24 inches, aligning with the expected output. It provides additional context by mentioning the measurement in millimeters and noting that other factors may influence the cover, which, while not in the expected output, does not contradict it. There are no factual inaccuracies or critical omissions, but the extra details slightly deviate from the expected brevity.
Q. Prompt: In the NEC which section deals with minimum cover requirement for cables and conduits. **A. Ground truth:** In the NEC, section 300.5 deals with minimum cover requirement for cables and conduits.	**Q. Prompt:** Is there any restriction in running conductors in parallel in the NEC.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025 doi:10.20944/preprints202507.0537.v1

A. Actual output: According to my knowledge based on the context provided, the minimum cover requirements for cables and conduits can be found in Section 300.5 of the NEC. This section specifies the minimum cover requirements for direct-buried cable, conduit or other raceways. Additionally, you may find further information regarding mechanical protection for direct-buried cables in sections 840.48 and 830.47 of the NEC. **Score: 0.8009300906486064** **Reason:** The actual output correctly identifies Section 300.5 of the NEC as dealing with minimum cover requirements for cables and conduits, aligning with the expected output. It also provides additional information about related sections, which is not required but does not contradict the expected output. There are no factual inaccuracies or omissions regarding the primary question, but the extra details, while accurate, are not necessary for the specific query.	**A. Ground truth**: NEC section 310.10(H) talks about conductors in parallel. **A. Actual output:** Yes, according to Section 310.10(H) of the NEC, parallel conductors are permitted to be installed in raceways, multiconductor cables or direct-buried single conductor cables. **Score: 0.5432697039021899** **Reason:** The actual output correctly identifies NEC section 310.10(H) as relevant to conductors in parallel, aligning with the expected output. However, it introduces additional details about installation methods that are not present in the expected output, which could be seen as extraneous information. There are no factual inaccuracies or contradictions, but the extra details could be considered an omission of focus on the core expected content.
Q. Prompt: Based on section 310.10(H) of the NEC, if I have two parallel #8 AWG conductors, what is my restriction on overcurrent protection. **A. Ground truth:** With two parallel #8 AWG conductors, the overcurrent protection should be such that the ampacity of individual conductors will not be exceeded if one or more of the parallel conductors become inadvertently disconnected. **A. Actual output:** According to Section 310.10(H) of the NEC, parallel conductors are allowed in raceways, multiconductor cables or direct-buried single conductor cables if they meet certain conditions: The total current-carrying capacity of the parallel circuits shall not exceed the ampacity rating of the raceway or overhead circuit. Based on the given context, if you have two parallel #8 AWG conductors, it is important to ensure that the total current-carrying capacity of both circuits does not exceed the ampacity rating of the raceway or overhead circuit.	**Q. Prompt:** I am trying to find the table in the NEC that deals with AC motors and associated current. **A. Ground truth:** Tables 430.248 through 430.250 reflect the typical and most used 4-pole and 2-pole induction motors (AC) in use. **A. Actual output:** According to the National Electrical Code (NEC), the section dealing with AC motors and their associated currents can be found in Article 440, which is titled “Motor Controllers.” Specifically, you may refer to Table 440.5 of Article 440 for information on the minimum full-load current required for a single-phase or three-phase AC motor

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025 doi:10.20944/preprints202507.0537.v1

Score: 0.2610665811531422 **Reason:** The actual output fails to specifically mention the restriction on overcurrent protection related to the scenario where one or more parallel conductors become disconnected, which is a key aspect of the expected output.	based on the nameplate rating of the motor. **Score: 0.20462199541570797** **Reason:** The actual output incorrectly identifies Article 440 and Table 440.5 as the relevant section for AC motors, while the expected output specifies Tables 430.248 through 430.250. This indicates a factual inaccuracy and a significant contradiction. Additionally, the actual output fails to address the specific tables mentioned in the expected output, leading to an omission of required information.

* A parsed version of the 2017 NFPA 70 – National Electric Code (NEC) was used for the above RAG implementation.

Several key observations emerged during the evaluation of the RAG-LLM pipeline using NEC-based queries:

1. *Semantic prompting and section identification:* The RAG-LLM pipeline demonstrated strong capability in retrieving and identifying the correct NEC sections relevant to a given question. However, optimal performance often required minor prompt reformulations to guide the language model semantically toward the appropriate retrieval vector. Prompts that aligned more closely with the phrasing or terminology found in the NEC tended to yield more accurate and relevant outputs, highlighting the importance of semantic alignment in user queries.
2. *Numerical table retrieval and chunking limitations:* The system exhibited average performance when answering questions that required retrieving numerical values from tabular data. In several cases, particularly those involving large or complex NEC tables, the retrieval process failed to return a coherent representation of the table. This behavior is attributed to the document chunking strategy—when large tables are split across multiple chunks, the model often receives incomplete context, resulting in incorrect or partially accurate responses. This fragmentation of tabular structure presents a known limitation in current RAG implementations.
3. *Challenges with multi-condition exceptions:* The RAG-LLM pipeline struggled with rules that involved multi-conditioned exceptions; this is common in NEC clauses that specify alternate requirements based on voltage, environment, or application type. These exceptions are often presented as enumerated or nested logic conditions, and the system had difficulty parsing and reasoning through the full set of conditions. As a result, responses sometimes omitted key qualifying criteria or misapplied the rule entirely. This highlights a current gap in the system's ability to interpret hierarchical exception logic within highly structured regulatory texts.

These observations suggest specific areas for refinement, particularly in document chunking, semantic prompting, and exception reasoning, which point toward the need for more advanced RAG strategies such as reranking based on a higher number of top-K results and contextual RAG strategies. These enhancements are discussed in detail in the subsequent section.

5. Enhancements to Traditional RAG-LLM Workflow

Building upon the limitations of the traditional RAG-LLM workflow outlined in the preceding section, the following section introduces targeted enhancements, namely reranking and contextual retrieval, that address identified gaps in relevance and specificity. While advanced in capability, these

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025 doi:10.20944/preprints202507.0537.v1

enhancements are readily implementable within the low-code N8N framework. For reproducibility and practitioner adoption, these advanced N8N workflow JSON files are provided and referenced in the "Data Availability" section.

5.1. Enhancements Based on Reranking

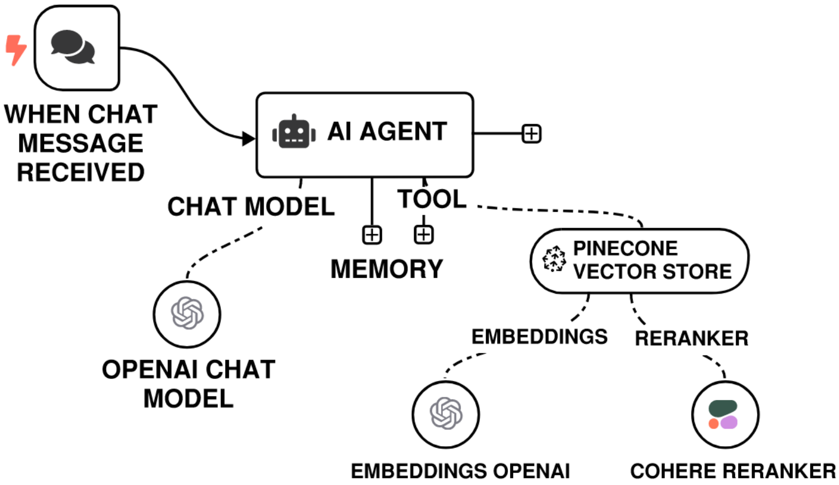
Reranking is a critical step in modern information retrieval and retrieval-augmented generation (RAG) pipelines, where a lightweight retriever first fetches a broad set of candidate documents, and a more sophisticated model subsequently reorders based on their true relevance to the query. Among reranking techniques, cross-attention-based reranking stands out for its ability to model fine-grained semantic alignment between the query and candidate passages. Unlike bi-encoder architectures, which independently encode queries and documents into fixed-length embeddings, cross-attention models (such as BERT or T5 used in a cross-encoder configuration) jointly process the query and document pair within a single transformer. This allows the model to compute token-level interactions, effectively attending to contextually relevant parts of each passage in relation to the query. The resulting representations yield significantly more accurate relevance scores, often leading to notable gains in retrieval performance, especially in tasks requiring nuanced understanding or disambiguation. While this approach incurs higher computational overhead—making it impractical for initial retrieval across large corpora—it is highly effective when applied to rerank a limited set of top-K candidates, providing a robust balance between precision and scalability.

From an implementation standpoint, reranking based enhancements can be achieved in an N8N workflow with relative ease. A vector database in N8N can be enabled to rerank results and connecting the reranking node to a reranking model such as Cohere Rerank 3.5, see Figure 4a. Behind the scenes, the Rerank API processes the input query by segmenting the associated document into smaller text chunks. Each chunk comprises the query along with a section of the document, and the chunk size is determined by the context length supported by the selected model. For instance, consider the following configuration:

- The model in use is rerank-v3.5, which supports a maximum context length of 4096 tokens, and
- The input query consists of 100 tokens, and,
- The document to be ranked is 10,000 tokens long, and
- Document truncation is turned off by assigning 'max_tokens_per_doc' a value of 10,000.

Under this setup, the API splits the document into three chunks, as illustrated in Figure 4b. The final relevance score assigned to the document corresponds to the highest score obtained across these chunks.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.



(A)

relevance_score_1 = <padding_tokens, query[0,99], document[0,3992]>
relevance_score_2 = <padding_tokens, query[0,99], document[3993,7985]>
relevance_score_3 = <padding_tokens, query[0,99], document[7986,9999]>

(B)

Figure 4. (a) N8N RAG-LLM workflow with Cohere Reranking. (b) Relevant scoring based on Cohere chunking on a 10,000 token document.

To assess the performance of various embedding models and rerankers, multiple retrieval tests were conducted. Table 3 presents the evaluation results using two standard information retrieval metrics: hit rate and Mean Reciprocal Rank (MRR). Three reranking configurations were compared:

1. Without any reranking (baseline),
2. Bge-reranker-base, and
3. Cohere V3.5 reranker.

With results documented in Table 3, the following conclusions can be drawn:

1. OpenAI + CohereRerank combinations consistently achieve the highest scores across both hit rate and MRR, positioning them as the top-performing setup.
2. Both CohereRerank and bge-reranker-base demonstrate consistent improvements across diverse embedding models, indicating their robustness and effectiveness in enhancing search quality regardless of the embedding backbone used.

Table 3. Open LLM leaderboard filtered by 'mid-range' models and sorted by weighted average of normalized scores from all benchmarks (as of June 2025). Note: boldface score indicates highest performance.

Embedding	**Without any reranking (baseline)**		**bge-reranker-base**		**Cohere V3.5 reranker**	
	Hit Rate	MRR	Hit Rate	MRR	Hit Rate	MRR
OpenAI	0.828	0.692	0.899	0.822	**0.932**	**0.877**
Voyage	0.766	0.650	0.818	0.777	0.891	0.830
Google-PaLM	0.815	0.676	0.892	0.752	0.916	0.865

The influence of rerankers cannot be overstated as they play a key role in improving the MRR for many embeddings, showing their importance in making search results better. In the next section the discussion would be directed towards another popular RAG enhancement technique using contextual retrieval.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

5.2. Enhancements Based on Contextual Retrieval Pipeline and its Advantages

In traditional Retrieval-Augmented Generation (RAG) pipelines, source documents are typically split into smaller text chunks to improve retrieval granularity and search efficiency. This chunking approach works well for many applications, particularly when relevant context is self-contained within each segment. However, in domains involving structured regulatory texts, such as electrical codes and standards, this method can introduce significant context fragmentation. For instance, consider the following query posed against a knowledge base containing the National Electrical Code (NEC):

> *"What is the minimum burial depth for direct-buried conductors under a parking lot?"*

A retrieved chunk might contain the following response:

> *"The minimum cover depth shall be 24 inches for direct-buried conductors."*

While this statement appears relevant, it lacks critical contextual qualifiers. The retrieved chunk may not explicitly include the fact that this requirement applies only to specific installation types, such as circuits rated over 0 to 600 volts and under specific locations like driveways or parking areas subject to vehicular traffic. If the installation condition or voltage rating is discussed in a preceding chunk, the answer produced by the RAG pipeline may be incomplete or misleading, even though the retrieval was technically successful. This illustrates a core limitation of traditional chunk-based RAG systems: individual chunks may not carry enough semantic or structural context to fully support precise, regulatory-compliant answers. In complex documents like the NEC, where applicability often depends on a combination of table values, conditional rules, and cross-referenced sections, this fragmentation can degrade both retrieval relevance and generative accuracy. These limitations motivate the need for contextual RAG approaches, which aim to preserve and reintroduce broader contextual frames during both retrieval and generation.

To address this, contextual RAG augments each chunk with its surrounding textual context (e.g., parent sections, headers, or preceding paragraphs) during embedding. This allows the retriever to maintain semantic continuity and structural fidelity. However, this comes at the cost of an increased token volume. Empirically, contextual RAG embeddings can require 2–3 times more tokens per chunk than traditional RAG approaches.

As token volume increases, careful model selection becomes essential, especially for real-time or cost-sensitive applications. For retrieval and summarization tasks that do not require advanced reasoning, it is preferable to select lightweight LLMs with:

- High token throughput (e.g., 200 - 300 tokens/sec),
- Low per-token cost (e.g., <$0.10 per million tokens), and
- Extended context windows (e.g., 1 million tokens).

These characteristics, summarized in Table 4, ensure that contextual RAG systems remain responsive and scalable, even as embedding and inference loads grow with richer document representations.

Table 4. Comparison of traditional vs. contextual RAG chunking strategies, highlighting trade-offs.

RAG strategy	**Avg token per chunk (or chunk + context)**	**Retrieval fidelity**	**Embedding cost**
Traditional RAG	~250–350 tokens	Medium	Low
Contextual RAG	~600–1000 tokens	High	Higher (unless models like Gemini 2.5 Flash

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025

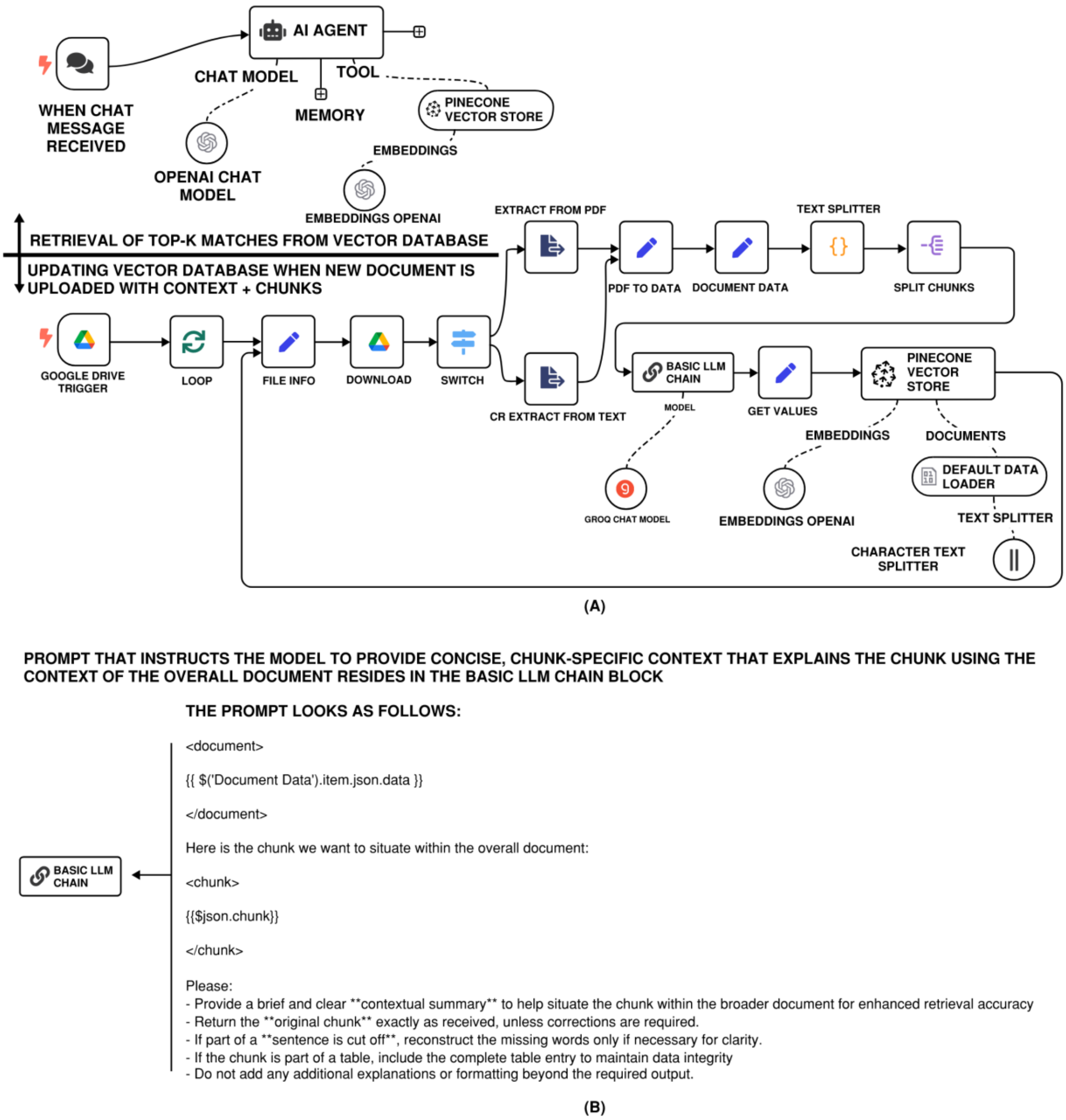
doi:10.20944/preprints202507.0537.v1

	are used, with low $/million token cost)

From a system architecture standpoint, the implementation of contextual RAG within an N8N workflow closely mirrors that of traditional RAG pipelines (recall Figure 3). The overall structure, illustrated in Figure 5a comprises of document ingestion, embedding generation, vector storage, retrieval, and language model invocation, remains fundamentally the same. However, contextual RAG introduces two key enhancements that differentiate it in both design philosophy and execution:

1. First, text splitting and chunking are performed more deliberately, with explicit control over chunk size, overlap, and structural boundaries. This ensures that semantically cohesive units, such as full table entries, complete regulatory exceptions, or paragraph-level logical constructs, are preserved as atomic retrieval units. This refinement is critical for maintaining the integrity of context during downstream retrieval and reasoning.
2. Second, instead of aggregating top-K retrievals into a single prompt, each chunk is individually passed to a "Basic LLM Chain" node within N8N. This node is configured with a well-crafted, structured prompt (user message), Figure 5b, that guides the LLM in evaluating each chunk's relevance and factual contribution to the original query. The design of this prompt is inspired by prompt templates found in contextual RAG applications such as those published by Anthropic [10]. The chain then filters or ranks responses from multiple chunks before synthesizing a final answer.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.



```
<document>
{{ $('Document Data').item.json.data }}
</document>
Here is the chunk we want to situate within the overall document:
<chunk>
{{$json.chunk}}
</chunk>
Please:
- Provide a brief and clear **contextual summary** to help situate the chunk within the broader document for enhanced retrieval accuracy
- Return the **original chunk** exactly as received, unless corrections are required.
- If part of a **sentence is cut off**, reconstruct the missing words only if necessary for clarity.
- If the chunk is part of a table, include the complete table entry to maintain data integrity
- Do not add any additional explanations or formatting beyond the required output.
```

Figure 5. (a) N8N RAG-LLM workflow with contextual retrieval system allowing for prepending chunk-specific explanatory context to each chunk before embedding. (b) Prompt that instructs the model to provide concise, chunk-specific context that explains the chunk using the context of the overall document.

The working of the contextual retrieval system is validated by comparing responses from both traditional and contextual retrieval systems to questions that would need the system to refer tabular data or exceptions to code sections from within the NEC with the results being docketed in Table 5.

Table 5. Validation of traditional versus contextual RAG-LLM responses using N8N workflows on query tasks based on the 2017 National Electrical Code (NEC). Score is computed using DeepEval's correctness with criteria [GEval] metric.

Q&A responses and scores with traditional RAG-LLM	Q&A responses and scores with contextual RAG-LLM
Q. Prompt: Based on the NEC, for solar photovoltaic (PV) system application, what is the correction factor that needs to be applied to PV cables for an ambient temperature of 55	**Q. Prompt:** Based on the NEC, for solar photovoltaic (PV) system application, what is the correction factor that needs to be applied to PV cables for an ambient temperature of 55

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

degree C, and the temperature rating of the conductor being 75 degree C rated. **A. Ground truth:** According to NEC Table 690.31(A), the applicable correction factor is 0.67. **A. Actual output from traditional RAG:** According to the National Electrical Code (NEC) 2017, specifically Table 310.15(B)(2)(a) (renumbered in later versions as Table 310.15(B)(1) or Table 310.15(B)(1) Adjustment Factors), the required correction factor is 0.70 for 75°C rated PV conductors in 55°C ambient conditions. **Score: 0.1981536822161493** **Reason:** The actual output provides a correction factor of 0.70 based on a different NEC table than the expected output, which specifies a correction factor of 0.67 from Table 690.31(A). This indicates a factual inaccuracy and a contradiction with the expected output. Additionally, the actual output omits the specific reference to Table 690.31(A), which is essential information from the expected output.	degree C, and the temperature rating of the conductor being 75 degree C rated. **A. Ground truth:** According to NEC Table 690.31(A), the applicable correction factor is 0.67. **A. Actual output from contextual RAG:** According to NEC Table 690.31(A), for a 75 °C rated conductor in an ambient temperature of 51–55 °C, the applicable correction factor is 0.67. **Score: 0.8918686352485045** **Reason:** The actual output is factually accurate and provides the correct correction factor of 0.67, as expected. It fully addresses the expected output by specifying the NEC table and the temperature range, which adds clarity. There are no contradictions, and the output aligns well with the context and requirements of the input. The only minor shortcoming is the slightly more detailed explanation than necessary, but it does not detract from the correctness. **Score improved over traditional RAG implementation.**
Q. Prompt: Based on NEC section 310, what is the adjustment factor for 4 current carrying conductor? **A. Ground Truth:** According to NEC Table 310.15(B)(3)(a), the applicable correction factor is 80%. **A. Actual Output from traditional RAG:** According to Section 310.15(3)(a) of the National Electrical Code (NEC), the adjustment factors for four current-carrying conductors are 1% and 2%. These factors are used to adjust the load diversity of the installation. **Score: 0.08819886431169416** **Reason:** The Actual Output incorrectly identifies the adjustment factors as 1% and 2%, which does not match the Expected Output's 80% correction factor. It also references the wrong section, 310.15(3)(a), instead of Table 310.15(B)(3)(a). This demonstrates factual inaccuracies and discrepancies with the Expected Output.	**Q. Prompt:** Based on NEC section 310, what is the adjustment factor for 4 current carrying conductor? **A. Ground Truth:** According to NEC Table 310.15(B)(3)(a), the applicable correction factor is 80%. **A. Actual Output from contextual RAG:** According to the National Electrical Code (NEC) 2017, specifically Table 310.15(B)(3)(a), the applicable correction factor as adjusted for ambient temperature if necessary is 80%. **Score: 0.7562603783294554** **Reason:** The actual output correctly identifies the adjustment factor as 80% according to NEC Table 310.15(B)(3)(a), aligning with the expected output. However, it introduces additional information about ambient temperature adjustment, which is not required by the input or expected output, slightly deviating from the expected response. **Score improved over traditional RAG implementation.**

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025 doi:10.20944/preprints202507.0537.v1

Q. Prompt: Based on section 310.10(H) of the NEC, if I have two parallel #8 AWG conductors, what is my restriction on overcurrent protection.	**Q. Prompt:** Based on section 310.10(H) of the NEC, if I have two parallel #8 AWG conductors, what is my restriction on overcurrent protection.
A. Ground truth: With two parallel #8 AWG conductors, the overcurrent protection should be such that the ampacity of individual conductors will not be exceeded if one or more of the parallel conductors become inadvertently disconnected.	**A. Ground truth:** With two parallel #8 AWG conductors, the overcurrent protection should be such that the ampacity of individual conductors will not be exceeded if one or more of the parallel conductors become inadvertently disconnected.
A. Actual Output from traditional RAG: According to Section 310.10(H) of the NEC, parallel conductors are allowed in raceways, multiconductor cables or direct-buried single conductor cables if they meet certain conditions: The total current-carrying capacity of the parallel circuits shall not exceed the ampacity rating of the raceway or overhead circuit. If the ungrounded service-entrance conductors are installed in parallel, they must be sized to match the ungrounded service-entrance conductor ampacity and rated voltage. Based on the given context, if you have two parallel #8 AWG conductors, it is important to ensure that the total current-carrying capacity of both circuits does not exceed the ampacity rating of the raceway or overhead circuit.	**A. Actual Output from contextual RAG:** According to NEC 2017, Section 310.10(H), for two parallel #8 AWG conductors, each conductor must be protected so that no individual conductor exceeds its ampacity.
Score: 0.2610665811531422 **Reason:** The actual output fails to specifically mention the restriction on overcurrent protection related to the scenario where one or more parallel conductors become disconnected, which is a key aspect of the expected output.	**Score: 0.8042020327293316** **Reason:** The actual output aligns well with the expected output by stating that each conductor must be protected to not exceed its ampacity, which is consistent with the requirement that the ampacity of individual conductors should not be exceeded. **Score improved over traditional RAG implementation.**

While contextual RAG represents a meaningful advancement in retrieval fidelity over traditional RAG-LLM (or standalone LLMs), both the traditional and contextual RAG techniques remain susceptible to hallucination, especially under the following conditions:

1. Complex or multi-layered prompts, where the query involves reasoning across multiple conditions or clauses.
2. Cross-referenced rules, common in regulatory and engineering texts, where one section refers to definitions, exceptions, or constraints located in another part of the document.

In these cases, even if the correct passage is partially retrieved, the language model may synthesize responses that are plausible but not fully aligned with the original source material. This issue arises because the generative component of the pipeline still relies on learned statistical patterns and does not inherently verify or enforce rule-bound reasoning.

For this reason, both traditional and contextual RAG should be regarded as tools to expedite the search and localization of domain-specific knowledge, rather than as definitive sources of truth. While these systems significantly reduce the cognitive and temporal burden of manual technical engineering code-book navigation, the final output must always be verified against the original text from the authoritative source - be it an engineering standard, regulatory code, or technical manual.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025 doi:10.20944/preprints202507.0537.v1

This verification step remains essential in high-assurance domains where regulatory compliance, safety, and precision are non-negotiable.

6. Advanced Implementation with Multi-Brain Knowledge Graph Based RAG

Knowledge graphs [11–13] offer an advanced paradigm for modeling relationships between concepts by structuring information as entities and their interlinked associations, rather than relying on high-dimensional vector embeddings. Unlike vector databases that encode semantic similarity through numerical proximity in an embedding space, knowledge graphs preserve explicit, interpretable relationships, such as hierarchical, causal, or functional links, through a graph-based schema. Each node represents a distinct concept or entity, while edges capture the nature of their interconnection, enabling nuanced reasoning, ontology-driven inference [14], and contextual disambiguation. Graph databases (e.g., Neo4j, RDF stores) serve as the underlying infrastructure, allowing for complex queries using graph traversal or SPARQL-like languages. This structure is particularly advantageous in domains requiring transparent decision paths, rule-based logic, or domain-specific ontologies, where relational integrity and explainability are critical. In addition to their structural clarity and interpretability, knowledge graphs provide unique analytical affordances that go beyond traditional embedding-based representations. These affordances enable the discovery of hidden patterns, support exploratory analysis, and reveal nuanced relationships within complex information spaces:

1. *Identification of structural gaps and blind spots:* As a more visual and relational representation of knowledge, knowledge graphs can expose under-connected or isolated nodes, that may indicate overlooked or weakly integrated concepts. These blind spots can be leveraged to generate novel insights by prompting new, contextually relevant connections between disparate ideas.
2. *Exploratory pathways for idea navigation:* Knowledge graphs facilitate intuitive exploration of conceptual relationships. By tracing how a given node connects to others across the graph, users can uncover indirect yet meaningful pathways between ideas, supporting hypothesis generation, interdisciplinary discovery, or refinement of conceptual frameworks.
3. *Revealing nuance through concept removal:* By algorithmically or manually removing dominant or highly connected nodes, knowledge graphs can surface latent structures and peripheral relationships. This approach highlights less obvious, contextually rich ideas that are often obscured by central concepts, enabling deeper interpretation and nuanced understanding of the information space.

Building on an understanding of the powerful features of knowledge graphs, one can integrate multiple knowledge graphs with an intelligent AI agent within a knowledge graph-based Retrieval-Augmented Generation (RAG) system. In this architecture, the AI agent dynamically (and in a non-linear manner) selects and traverses one or more connected knowledge graphs based on the input prompt. By leveraging the graph structure for semantically rich retrieval, the system can extract contextually relevant insights, uncover implicit relationships, and provide more precise, interpretable responses grounded in structured domain knowledge. This approach holds significant promise for engineering research applications, where complex, interrelated knowledge domains benefit from structured representation and contextual retrieval. As a demonstration, the authors developed three functionally distinct knowledge graphs, each tailored to a specific thematic domain. The specific thematic domain here broadly consists of an electrical engineering issue where a researcher is wanting to leverage their own research, and published IEEE and CIGRE technical reference documents and manuscripts on the subject of ‘grid resiliency enhancement’. Three custom knowledge graphs (referred to as brain 1, 2, and 3) are developed for this purpose:

1. Brain #1 encapsulates the authors’ own research outputs, including published manuscripts [15,16] and internal technical reports focused on grid maintenance planning, resiliency metrics, substation floor design, and fire risk assessments.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025 doi:10.20944/preprints202507.0537.v1

2. Brain #2 houses IEEE technical reports such as IEEE TR83 [17] and selected published manuscripts [18–20] from the IEEE Xplore Digital Library, dedicated to resiliency frameworks, analytical methods, and performance metrics relevant to the electric power sector.
3. Brain #3 aggregates select CIGRE technical reports [21] and manuscripts [22], specifically curated to serve as a domain expert on sustainable strategies for enhancing substation resilience and reliability, including the adoption of eco-friendly equipment and remote maintenance systems.

The architecture of this knowledge graph-based RAG implementation is shown in Figure 6a, with all three knowledge graphs generated and hosted on an InfraNodus platform [23]. Figure 6b provides a visual representation for one out of the three knowledge graphs in use. Similar to the previous implementations of the traditional RAG architecture and the contextualized RAG architecture, this knowledge graph-based RAG implementation is developed using the N8N platform and is available for others to use; see the data availability section of the manuscript. Depending on the nature and specificity of the research prompt, the AI agent can query one, several, or all these interconnected “brains,” thereby retrieving the most contextually relevant insights and maximizing knowledge utility across diverse yet related knowledge domains. Queries and knowledge graph-based responses are documented in Table 6. As one may observe, the underlying AI agent dynamically selects the knowledge graphs that it needs to tap into.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025 doi:10.20944/preprints202507.0537.v1

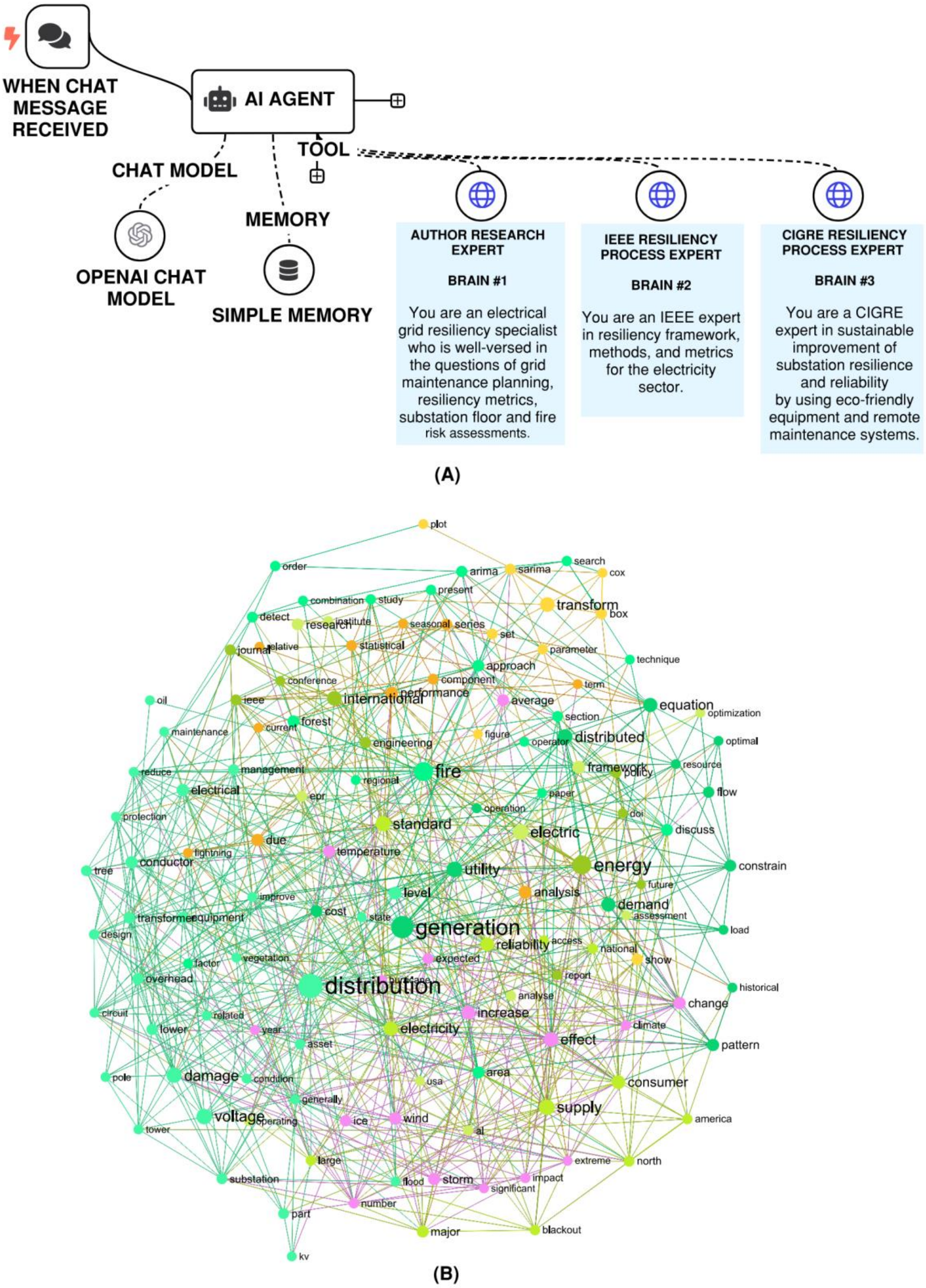


Figure 6. (a) N8N RAG-LLM workflow with multiple custom knowledge graphs, each housing unique knowledge. (b) A visualization of the 'author research expert' (brain 1) knowledge graph.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Table 6. Testing the working mechanic of the knowledge graph-based RAG implementation with the AI agent intelligently tapping the necessary dynamic knowledge graphs (brains).

Prompt into knowledge graph-based RAG	Brains which the AI agent consulted	Response from the knowledge graph-based RAG agent
Q. Prompt: Referring to the IEEE technical reference documents and the author's own research, can you summarize very briefly how fast trip relay settings can help in high fire threat areas.	Brain 1 and Brain 2 (see Figure 6a)	**A. Actual Output from knowledge graph-based RAG agent:** Fast trip relay settings contribute significantly to enhancing grid resiliency in high fire threat areas. According to both IEEE documents and research on grid resilience: Rapid fault detection and isolation: Fast trip relay settings quickly identify and isolate faults along power lines, which is crucial in preventing small issues from escalating into larger problems that could ignite wildfires. Proactive risk mitigation: By setting sensitive thresholds for detecting faults, relays act as a proactive defense mechanism. They mitigate the risk of electrical faults that could lead to fire ignition, thereby aligning with a risk mitigation strategy that addresses specific hazards prevalent in high fire risk zones. System reliability: This approach not only prevents potential fires but also enhances the overall system's reliability and safety, maintaining electric continuity and preventing outages that could compound firefighting efforts. Integrating such relay settings reflects a focused effort on prevention and protection, key aspects of a resilient grid especially in regions vulnerable to wildfires.
Q. Prompt: Referring CIGRE documents and the author's own research, can you list briefly what are the be benefits of SF6 alternative circuit breaker.	Brain 1 and Brain 3 (see Figure 6a)	**A. Actual Output from knowledge graph-based RAG agent:** The benefits of using SF6 alternative circuit breakers, specifically those using vacuum and synthetic air, include: Environmental sustainability: SF6 alternative circuit breakers, such as those using vacuum technology and synthetic air, significantly reduce environmental impact by eliminating greenhouse gas emissions associated with SF6, which has a high Global Warming Potential (GWP). Improved safety: These alternatives enhance safety by simplifying the gas recovery and filling processes, and they avoid the release of harmful gases into the environment. Cost-effective maintenance: The vacuum technology extends maintenance cycles, reducing overall costs and increasing operational efficiency, as noted in innovative

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

	applications that align with sustainable development goals.

Based on a series of prompting interactions with the knowledge graph-based RAG system, the following conclusions were made:

1. *Higher API usage cost for knowledge graph generation:* From an implementation standpoint, generating a knowledge graph from source material (e.g., PDFs or structured text) incurs significantly higher API consumption compared to traditional vector database ingestion. For instance, while basic text embedding may consume 1–2 tokens per word, the creation of a knowledge graph—particularly those involving relation extraction, entity disambiguation, and ontology alignment—can require 5–10 × more tokens per document. This increased cost reflects the deeper semantic parsing and relationship mapping inherent to graph construction.

Recommendation to reduce API usage: The recommendation in Figure 7 outlines a cost-efficient, stepwise strategy for implementing RAG systems, starting with traditional methods and escalating to contextual or knowledge graph-based RAG only when deeper semantic understanding or domain-specific relational reasoning is required.

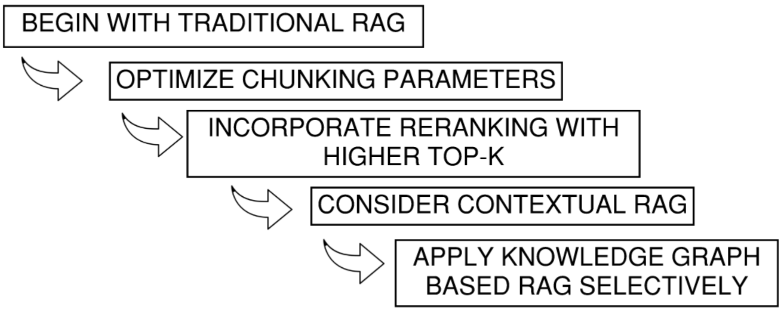

BEGIN WITH TRADITIONAL RAG
OPTIMIZE CHUNKING PARAMETERS
INCORPORATE RERANKING WITH HIGHER TOP-K
CONSIDER CONTEXTUAL RAG
APPLY KNOWLEDGE GRAPH BASED RAG SELECTIVELY

Figure 7. A multi-step guide to reduce API usage for RAG implementation.

2. *Limitations with numerical tabular data:* Knowledge graphs are not well-suited for representing dense numerical tabular data, as they excel at modeling conceptual and relational structures rather than high-volume matrix-like values. Hence using knowledge graphs for rich tabular content, such as the NEC or other NFPA standards would be limiting. Through the workflow, it was observed that when numerical tables are forced into a graph format, granularity is lost, and retrieval becomes inefficient; numeric precision, row-wise dependencies, and columnar statistics do not translate naturally into entity-relation triples, limiting effective graph-based inference in such contexts.
3. *Expanded exploratory potential with multiple knowledge graphs:* A key advantage of using multiple specialized knowledge graphs is the dynamic expansion of the knowledge space accessible to the AI agent. Depending on the prompt, the agent can selectively or concurrently traverse these graphs, uncovering deeper and more diverse interconnections. This multi-graph strategy enhances exploratory analysis, allowing users to follow conceptual pathways across different but related domains, uncover latent insights, and generate novel hypotheses grounded in a broader knowledge base.
4. *Other observations:* Some additional observations that were made are docketed as follows:
    a. Traditional vector-based RAG loses structural information when chunking documents, while such hierarchies and relationships are preserved in knowledge graph-RAG.
    b. Knowledge graph-RAG were better able to support complex reasoning across multiple facts, such as exceptions to certain engineering code sections, or supplementary discussions on a particular subject within a research paper.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

7. Summary and Future Work

This manuscript demonstrates that with the aid of powerful AI automation platforms such as N8N, robust Retrieval-Augmented Generation (RAG) pipelines can be constructed and deployed with minimal coding overhead. Engineering professionals and researchers can now build highly functional RAG systems that seamlessly retrieve and process domain-specific documents. Document repositories can be indexed using either vector databases or knowledge graph structures, enabling flexible and context-aware querying through low-code workflows. While vector-based RAG remains the mainstream approach due to its computational efficiency and ease of deployment, knowledge graph-based RAG offers deeper semantic structuring and superior interpretability, particularly valuable for navigating interconnected technical concepts across engineering standards.

At present, the widespread adoption of knowledge graph-augmented RAG is limited by its computational complexity and relatively high API costs for graph generation and reasoning. However, as graph processing APIs become more cost-effective and efficient, knowledge graph-based retrieval is expected to gain momentum and may eventually surpass vector-based methods in both precision and domain relevance. This approach holds particular promise in high-integrity fields such as electrical engineering, where understanding contextual relationships and dependencies between technical terms, clauses, and procedures is crucial. The use of tools such as Infranodus in conjunction with N8N exemplifies the practical feasibility of this direction, paving the way for advanced semantic querying within document-rich environments.

Looking ahead, several areas offer opportunities for further development. These include:

1. Expanding support for multi-modal retrieval, such as integrating diagrams and structured tables into the RAG workflow,
2. Developing more efficient chunking and indexing strategies to reduce token overhead,
3. Enabling real-time collaboration and feedback loops between human experts and RAG systems, and
4. Improving support for multilingual engineering documents. Continued refinement of low-code frameworks and better benchmarking across diverse engineering datasets will further strengthen the practical utility of RAG-LLM systems in technical disciplines.

Author Contributions: S.G. supervised this study, ran interviews to generate appropriate prompts, contributed to the N8N workflow generation, and wrote parts of Section 1, Section 2, Section 4, Section 5, and Section 6. G.M performed the necessary benchmarking studies, contributed to data analysis, and wrote parts of Sections 1, 3, and 7. All researchers contributed to the review, editing, proofreading, and revision of this paper. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The N8N templates generated as part of this manuscript can be accessed on the project's GitHub page, along with Python code for document parsing. A Python notebook is also available for scoring LLM generated outputs versus ground truth. Github link: https://github.com/sghosh27/Low-Code-RAG-LLM-Framework-for-Context-Aware-Querying-in-Electrical-Standards-Design-and-Research

Acknowledgments: The views, thoughts, opinions, and conclusions made in this manuscript are solely those of the authors and don't necessarily reflect the view of the authors' employer, committee, or other groups or individuals. The authors are solely responsible for the content and accuracy of the technical manuscript.

The authors would like to thank Dmitry Paranyushkin, founder of InfraNodus for his seminal work on AI knowledge base and making the platform available to the public and research community.

Conflicts of Interest: The authors declare no conflicts of interest.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Abbreviations

The following abbreviations are used in this manuscript:

IEEE	Institute of Electrical and Electronics Engineers
LLM	Large Language Models
NEC	National Electric Code (also known as NFPA 70)
NFPA	National Fire Protection Association (US)
RAG	Retrieval-Augmented Generation

References

1. J. Li, X. Cheng, W. X. Zhao, J.-Y. Nie and J.-R. Wen, "HaluEval: A large-scale hallucination evaluation benchmark," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.
2. D. Roustan and F. Bastardot, "The clinicians' guide to large language models: A general perspective with a focus on hallucinations," *Interactive Journal of Medical Research*, 2025.
3. T. Zhang, L. Qiu, Q. Guo, C. Deng, Y. Zhang, Z. Zhang, C. Zhou, X. Wang and L. Fu, "Enhancing uncertainty-based hallucination detection with stronger focus," *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 915-932, 2023.
4. M. Omar, V. Sorin, J. D. Collins, D. Reich, R. Freeman, N. Gavin, A. Charney, L. Stump, N. L. Bragazzi, G. N. Nadkarni and E. Klang, "Large language models are highly vulnerable to adversarial hallucination attacks in clinical decision support: A multi-model assurance analysis," MedRxiv, 2025.
5. S. Gopi, D. Sreekanth and N. Dehboz, "Enhancing engineering education through LLM-driven adaptive quiz generation: A RAG-based approach," in *IEEE Frontiers in Education Conference (FIE)*, Washington, DC, US, 2024.
6. L. Siddharth and J. Luo, "Retrieval augmented generation using engineering design knowledge," *Knowledge-Based Systems*, vol. 303, 2024.
7. J. Superbi, H. Pereira, E. Santos, L. Lattari and B. Castro, "Enhancing large language model performance on ENEM math questions using retrieval-augmented generation," in *Proceedings of the XVIII Brazilian e-Science Workshop (BreSci)*, 2024.
8. Y. Chen, Q. Fu, Y. Yuan, Z. Wen, G. Fan and D. Liuet, "Hallucination detection: robustly discerning reliable answers in large language models," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2023.
9. Huggingface, "Open LLM Leaderboard," 2025. [Online]. Available: https://huggingface.co/spaces/open-llm-leaderboard/open_llm_leaderboard#/. [Accessed 2025].
10. Anthropic, "Introducing Contextual Retrieval," 09 2024. [Online]. Available: https://www.anthropic.com/news/contextual-retrieval. [Accessed 06 2025].
11. S. Ji, S. Pan, E. Cambria, P. Marttinen and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition, and applications," *IEEE Transactions on Neural Networks and Learning Systems (*, vol. 33, no. 2, pp. 494-514, 2022.
12. C. Peng, F. Xia, M. Naseriparsa and F. Osborne , "Knowledge graphs: Opportunities and challenges," *Artificial Intelligence Review*, vol. 56, pp. 19071-13102, 2023.
13. S. Tiwari, F. N. Al-Aswadi and D. Gaurav , "Recent trends in knowledge graphs: theory and practice," *Soft Computing*, vol. 25, p. 8337–8355, 2021.
14. K. Baclawski, E. S. Chan, D. Gawlick, A. Ghoneimy, K. Gross, Z. H. Liu and X. Zhang , "Framework for ontology-driven decision making," *Applied Ontology*, vol. 12, no. 3-4, 2017.
15. S. Ghosh and P. Suryawanshi , "Enhancing grid resiliency in high fire or flood risk areas: Integrating protective relay settings, broken conductor detection, and grid hardening for climate-induced event preparedness," *Journal of The Institution of Engineers (India): Series B*, vol. 106, pp. 393-405, 2024.
16. S. Ghosh and S. Dutta, "A comprehensive forecasting, risk modelling and optimization framework for electric grid hardening and wildfire prevention in the US," *International Journal of Energy Engineering*, vol. 10, no. 3, pp. 80-89, 2020.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 July 2025

doi:10.20944/preprints202507.0537.v1

17. A. Bose, S. Brown, B. Chalamala, D. Immerman, A. Khodaei, J. Liu, D. Novosel, A. Paaso, F. Rahmatian, J. R. Aguero and M. Vaiman, "Resilience framework, methods, and metrics for the electricity sector," IEEE Power & Energy Society (PES-TR83), 2020.
18. F. Safdarian, J. L. Wert, D. Cyr and T. J. Overbye, "Power system resiliency and reliability issues from renewable resource droughts," in *IEEE Kansas Power and Energy Conference (KPEC)*, Manhattan, Kansas, US, 2024.
19. S. Bhattarai, A. Sapkota and R. Karki, "Analyzing investment strategies for power system resilience," in *IEEE Power & Energy Society General Meeting (PESGM)*, Denver, Colorado, US, 2022.
20. A. Tabassum, S. Lee, N. Bhusal and S. Chinthavali, "Power outage forecasting for system resiliency during extreme weather events," in *IEEE International Conference on Big Data (BigData)*, Washington, DC, US, 2024.
21. E. Ciapessoni, D. Cirio, A. Pitto, M. Van Harte and M. Panteli, "Power System Resilience: definition, features and properties," *CIGRE Science and Engineering,* vol. CSE030, 10 2023.
22. R. Itotani, K. Sadahiro, M. Tokai, H. Hama, K. Sugino and M. Takeda, "Sustainable improvement on substation resilience and reliability by using eco-friendly equipment and remote maintenance systems," in *CIGRE Paris Session*, 2024.
23. D. Paranyushkin, "InfraNodus: Discover What Everyone Else Is Missing," 2024. [Online]. Available: https://infranodus.com/. [Accessed 07 2025].

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

© 2025 by the author(s). Distributed under a Creative Commons CC BY license.