

Retrieval augmented generation with LLMs for enterprise proposal automation

Mrs. Bakiyalakshmi.S¹, Sanjay C² and Sriram D³

¹Dept of CSBS, Rajalakshmi Engineering College, India bakiyalakshmi.s@rajalakshmi.edu.in

²Dept of CSBS, Rajalakshmi Engineering College, India sanjaychandrasedkar16@gmail.com

³Dept of CSBS, Rajalakshmi Engineering College, India sriramdhayan04@gmail.com

Abstract. Enterprise proposal writing as a response to RFPs is a very grave time-consuming undertaking since you must read and understand a stack of documentation correctly to ensure that everything is per the requirements. Of course, Large Language Models (LLMs) are able to generate fluent text, but in the context of depending solely on them they tend to produce either hallucinatory text or fail to pinpoint context as they are ungrounded. This paper demonstrates a Retrieval-Augmented Generation (RAG) system, which automates the process of proposal drafting in an enterprise by consuming docs and extracting text, chunking text into semantically meaningful objects, generating dense vectors, and performing similarity-based retrieval to provide the LLM with a real-world context. When a system runs, it retrieves the pertinent elements of a vector database in real-time and inserts them into a structured prompt and the LLM then runs, enhancing factual accuracy and maintaining the context tight. It cooperates with large API-based structures and smaller local ones, which means that you can make comparisons on performance, the efficiency of context handling, and the ability to deploy anything. Our experiments demonstrate that relevance and reduction in hallucinations are increased with the addition of retrieval over and above generating text blindly, and it is a scalable, modular means of putting AI to work on writing enterprise proposals.

I. INTRODUCTION

Request for Proposals (RFPs) are vital in organizations in the various industries when they need to find vendors, partners or service providers. The entire process of assembling such proposals is beyond time-consuming, and consumes a lot of resources, as you need to coordinate a large number of teams, process a large amount of information and ensure you are satisfying every single requirement of the client. The old-fashioned methods of writing and reading proposals are connected with the repetitive manual labor, excessive amount of miscommunication, and lack of time to meet the deadline, which can considerably reduce your chances to win business in fact. The development of AI and collaborative technologies eventually provides us with an opportunity to overhaul this entire process. Automated document readers can identify the requirements, highlight areas of ambiguity and sort out reusable information, whereas smart editing software can replicate the text, hint at better ways to do things, and make the entire whole look consistent across the pages. Teamwork aspects such as real-time editing, version control, and role-based allocation of tasks provide teams, including cross-functional units of legal and finance to the technical experts, with a more organized and efficient

approach to work. The suggested system introduces a smart editor combining automation and knowledge management to simplify the whole process behind the RFP.



Figure 1 : System Workflow

Document intake and content strategy, drafting, compliance checks and subsequent submission all reduce manual efforts and increase accuracy, speed and the quality of the proposal as a whole. In addition, with a knowledge base of past proposals, references and case studies stored, the system will continue to learn and

develop to provide organizations with competitive advantage in bagging new business opportunities.

II. LITERATURE REVIEW

Earlier studies on automated RFP response generation have largely concentrated on knowledge-based and model-driven frameworks to convert organizational knowledge into structured repositories to aid in the draft proposal composition. Rajbhoj et al. [1] proposed a system that utilizes structured knowledge representations and query-based retrieval to generate RFP responses and showed step and percent improvements in the precision of retrieval and drafting efficiency. Following the same line of thought, Nistala et al. [2] provided industrial experiences of the deployment of AI-enabled and model-driven proposal development systems in an enterprise setting and stated the advantages of reusable content modules and the difficulties of maintaining domain-specific knowledge bases on a large scale. Similar studies by Convertino et al. [8] highlighted the significance of collaborative reuse and organizational learning in proposal writing, which they said developed RFP as a knowledge intensive, multi stakeholders process that necessitated organized support structures. All these works prove the importance of systematization knowledge management and content reuse in proposal automation of enterprises.

In addition to generation, automated scoring and evaluation of responses on RFP have also been explored. Maji et al. [4] introduced an explainable deep learning model to rate RFP submissions, finding semantic patterns that are associated with the results of evaluation. Furthermore, Adhikari [12] has also investigated the proposal evaluation models based on learning that can learn subjective assessment criteria leveraging the linguistic and structural features. These systems seek to minimize the work of the reviewer and enhance consistency but rely on large and high quality labeled datasets and domain adaptation. Besides, the work on the lexical and semantic reuse detection by MacLaughlin et al. [5] offers background methods of detecting content overlaps and transfer of knowledge that are essential in the massive process of proposal drafting and evaluation.

Recent developments of generative AI-based response automation have taken the place of traditional studies with large language models (LLMs) emerging. In their study on Retrieval-Augmented Generation (RAG) to automate telecom RFP, Bilal and Astrom [6] found that retrieval-grounded generation brought a markedly better contextual relevance and a reduced number of hallucinations. A detailed overview of the RAG architectures was presented by Gao et al. [7], showing how retrieval elements address the limitations of the standalone LLMs, such as outdated knowledge and a deficit of domain grounding. Subdomain-specific

adoptions e.g. QA-RAG to regulatory compliance processes [11] demonstrate how retrieval-integrated generation can be used to improve traceability and factual reliability in compliance-sensitive applications. This evidence supports the suitability of RAG models in relation to enterprise proposal writing.

LLM has also been studied as complementary research on how to deal with structured and tabular data that often appear in RFP specifications. A study by Sui et al. [3] considered the performance of LLM on structured tabular input, and found out that it was limited to the reasoning of complex tabular input. Nguyen et al. [15] also investigated the case of the structured data generation under LLM, which can theoretically produce realistic tabular generation however, the logical consistency can be quite challenging. Research on generation-based self-training [9] and quantization approaches to the deployment of LLM [10] point to the developments in efficiency and scalability that are the key to the implementation in a realistic enterprise setting.

Organizationally and process-wise, Edwards [13] talked about the implementation of the knowledge management systems into the business processes and how AI-based solutions should be aligned to the processes of knowledge management in the enterprise. Thelijjagoda and De Silva [14] analyzed the governance aspects of dealing with ICT RFPs processes, which supported the importance of ordering validation and compliance controls in automated systems. Taken together, these works suggest that notwithstanding the important progress that has been achieved so far in the domain of proposal automation, scoring, retrieval augmentation, and the deployment of LLMs, the creation of a single system that will combine retrieval-grounded generation, structured prompt control, compliance alignment, and interactive editing is a still open field of application research. The proposed framework handles this gap by integrating the semantic retrieval, controlled generation, and enterprise-focused document management into a single RFP automation pipeline.

III. METHODOLOGY

A. System Overview

The suggested system is a Retrieval-Augmented Generation (RAG) model, which is aimed at automating the process of drafting proposals within the enterprise based on the Requests for Proposals (RFPs). It allows users to post RFP files, retrieve and semantically index their contents and produce structured response proposals based on contextual information retrieved. Combining document processing, semantic retrieval using vectors, structured prompt engineering, and large language model (LLM) generation, the system also makes the outputs context-sensitive, coherent, and consistent with the needs of the sources. The architecture can be deployed to a

variety of computational environments by supporting high-capacity API-based models and lightweight local models.

B. System Architecture

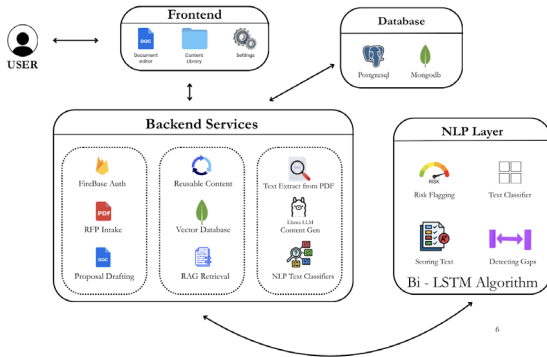


Figure 2 : System Architecture

This system is architected based on a modular Retrieval-Augmented Generation (RAG) model (combining document processing, semantic retrieval, and controlled large language model (LLM) inference). It has four main layers, namely frontend interface, backend services, vector retrieval infrastructure, and LLM inference. The interface (frontend), which is designed in React, is an interactive document editor that allows uploading RFPs, previewing the content, creating content with AI assistance, and editing the content with automatic saving. The back-end is developed on Fast API and handles PDF ingesting, text extraction, chunking, and embedding generation. A locally hosted embedding model is used to extract text and partition it into manageable segments and transform it into dense vectors. The metadata and these embeddings are kept in a Pinecone vector database to make them easily looked up using similarity, and the PDF files and the contents of the editors are stored in MongoDB to manage durable storage. The aim of the user during proposal generation is to compare the objective of the user with stored vectors by use of cosine similarity search which helps in retrieving the most relevant contextual segments. The context retrieved is integrated into an organized prompt template imposing predefined proposal sections and stylistic restrictions. This cue is then released to the LLM to generate responses. The architecture supports high capacity API based model and lightweight local model so that it could be deployed flexibly and compared. The architecture is based on the concept generation that provides contextual fit, less hallucination, and scales proposal automation on an enterprise level by basing generation on recovered RFP content.

C. Retrieval Augmented Generation Framework

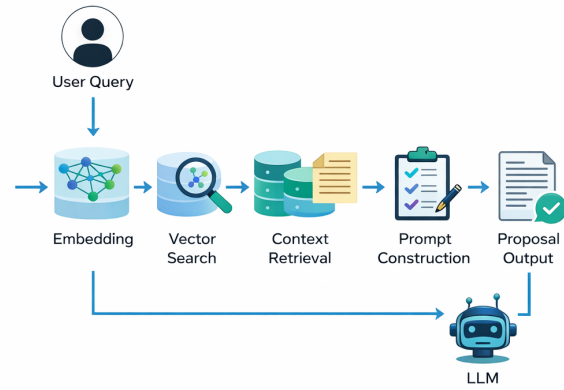


Figure 3: RAG Pipeline Flow

The suggested system uses a Retrieval-Augmented Generation (RAG) framework to secure the context-based drafting of a proposal. When uploaded, the contents of RFP files are broken down and divided into semantic units of a fixed size. They then convert every chunk to a dense embedding with the help of an embedding model trained in advance and create an index of the resulting vectors in a vector database to facilitate similarity-based retrieval. When generating, the purpose of the user gets integrated and compared with vectors stored under the cosine similarity search to get the top-k most relevant document segments. This retrieved segments are put together to make a context grounding layer. This situation is then combined into a formatted prompt template which imposes pre-established sections of proposals, stylistic restrictions, and rule-based restrictions. A large language model (LLM) then infers on the constructed prompt to generate a structured proposal committed to both the intent of the user and the content retrieved in the RFP. The framework minimizes hallucination and increases factual consistency by conditioning generation on semantically retrieved context, which is better than unconditioned LLM-based drafting.

D. Model Configurations and Design Choices

The system will facilitate the high capacity cloud-based inference and lightweight local deployment to decide the tradeoffs between the quality of generated and the computational efficiency. An API-based LLM (Llama 3.3 70B) is required to apply the inferences to a large scale due to more impressive context processing power, higher coherence, and enhanced instruction-following performance. This is organized in such a manner by benchmarking the best possible quality of output in case of retrieval-grounded prompting. Simultaneously with this a localized version of TinyLlama (1.1B) model is added to assess the potential to work within resource-restricted environments. Even at a much smaller model size, the local model has the advantage of being capable of inferring on a CPU, albeit with parameter trimming under control, at the expense of generative depth. The dense embedding model is trained in semantic

indexing to convert text fragments to get fixed-dimensional representation of vectors. The same embedding model of the document chunks and user queries is used to obtain the consistency of representation in the shared embedding space. The embeddings are stored in a serverless vector database which is configured to provide cosine similarity search. The rationale behind the use of cosine similarity lies in the fact that in high-dimensional semantic spaces, it is successful in calculating the angular distance, and that is why it is the perfect similarity as compared to normalized embeddings. The contextual completeness and prompt length limitation is sandwiched with the aid of top-k retrieval strategy. When small bits of extremely relevant chunks are recalled they reduce noise injection whilst preserving semantic grounding.

Table 1 : Component to Configurations

Component	Configuration	Rationale
LLM (API)	Llama 3.3 70B	High-quality generation, benchmarking
LLM (Local)	TinyLlama 1.1B	Resource-efficient deployment
Vector DB	Pinecone (cosine metric)	Scalable nearest-neighbor search

The moderate size of k values is sufficiently large to give sufficient context to span model context windows without being too numerous. The net impact of such design choices includes, but is not limited to, the fact that these designs are scalable and modular RAG pipeline, and that the model capacity, latency, and output fidelity can be compared.

IV. Experimental Evaluation

A. Experimental Setup

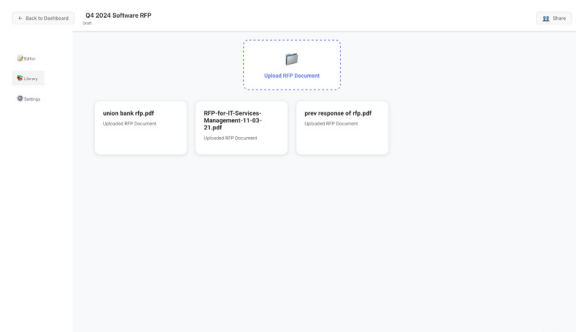


Figure 4: 3 RFP Documents in content library

The given framework has been compared with the set of the 3 enterprise RFP documents collected within the IT Services Industry where the size of the documents is approximately 70 -100 pages. Each document passed through all the steps of the pipeline that included text extraction, semantic chunking, embedding generation, and vectors indexing. Proposals were responded to in three environments that incorporated (i) standalone LLM (no retrieval) which was the baseline; (ii) retrieval-augmented generation on an API-based LLM (Llama 3.3 70B); and (iii) retrieval-augmented generation on a locally downloaded TinyLlama (1.1B) model. The qualitative and semi-quantitative analyses of the outcome were performed based on the contextual relevance, structural adherence to the predetermined parts of the proposal and hallucination reduction. The evaluation was performed based on the 2 reviewers who were conversant with the field on a scale of 5 scoring structure that they utilized to establish the concurrence with source requirements and factual consistency.2

B. Results and Analysis

The standalone LLM system demonstrated well-structured and fluent text generation but generalized the statement and baseless assumptions are common at a reach of around 40% hits. Outputs based on particular requirements of the RFP were not always operationalized at a baseline, but structurally coherent. Comparatively, the retrieval-augmented settings were quite helpful in facilitating contextual congruity via semantically retrieved document fragment conditioning on generation.

```

{
  "system": "You are a professional enterprise sales proposal writer.",
  "style": "Formal, concise, persuasive, enterprise-grade.",
  "structure": [
    "Executive Summary",
    "Client Understanding",
    "Proposed Solution",
    "Implementation Plan",
    "Pricing & Commercials",
    "Why Us",
    "Conclusion"
  ],
  "rules": [
    "Do not hallucinate numbers",
    "Use provided context only",
    "Use confident but professional tone"
  ]
}
    
```

Figure 5 : Prompt to write Proposals

The API based LLM showed the relevance score with the highest relevance scores and the proposals were close to the extracted requirements and with the estimated rating of approximately 30, the hallucinated claims significantly reduced. The TinyLlama system was structurally homogenous and was tolerably contextually grounded, despite small-scale linguistic regression and incomplete coverage of requirements due to the size of the model and the size of the context window.

Table 2 : Model Evaluation Results

Model	Retrieval	Hallucination	Contextual Relevance
LLM only	No	High	Moderate
RAG + API	Yes	Low	High
RAG + TinyLlama	Yes	Moderate	Moderate-High

```
f[assistant]> Executive Summary
Our company, [Company Name], offers a unique AI-powered document automation platform that drastically improves the efficiency and productivity of businesses. Our platform eliminates manual processes, saving time and cost, while enhancing customer service and reducing errors.

Client Understanding
Our platform automates a variety of document-related tasks, including document creation, data extraction, and sharing. By streamlining the document-related workflows, businesses can reduce labor costs while increasing productivity.

Proposed Solution
Our platform offers a fully automated document-related workflow system. The system utilizes a combination of natural language processing (NLP) and machine learning (ML) algorithms to identify and extract relevant data from documents.

Implementation Plan
Our implementation plan includes the following steps:
1. Conduct a thorough needs analysis to identify the specific document-related tasks that require automation.
2. Design and develop the automation platform using a cloud-based platform.
3. Train the platform's algorithms using a custom dataset of relevant documents.
4. Integrate the platform with the business's existing systems and workflows.
5. Conduct pilot trials with small-scale businesses to test the platform's effectiveness.
6. Launch the platform
```

Figure 6 : TinyLlama Output

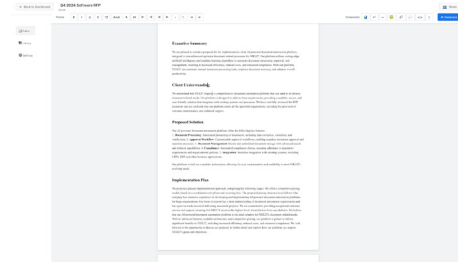


Figure 7: Groq Output

V. CONCLUSION

The modern article offered a framework of proposal automation in the enterprise with Retrieval-Augmented Generation (RAG) that involves document processing, semantic retrieval, and controlled large language model inference to improve the contextualization of RFP response drafting. The system also decreased hallucinations and structural adherence by a significant margin as compared to standalone LLM generation through embedding and indexing RFP content in a vector database and conditioning generation on context retrieved. A more detailed comparative study of a high capacity API-based model and a lightweight local model revealed the trade off between quality production and deployment performance under different conditions of computation, latency constraints, scalability, and resource availability. Overall, the proposed architecture can provide a flexible and scalable system of AI-assisted enterprise proposal generation, which can make a pragmatic contribution to practical NLP systems in business automation, organizational knowledge reuse, intelligent decision-support processes, and overall development of retrieval-grounded generative systems in enterprise-scale digital transformation projects.

REFERENCES

[1] P. Nistala, A. Rajbhoj, V. Kulkarni, S. Noronha, A. Joshi, An industrial experience report on model-based, AI-enabled proposal development for an RFP/RFI. *Sci. Comput. Program.* **233**, 103058 (2023). <https://doi.org/10.1016/j.scico.2023.103058>

[2] A. Rajbhoj, P. Nistala, V. Kulkarni, G. Ganesan, A RFP system for generating response to a request for proposal. In *Proceedings of the ACM Conference*, pp. 1–9 (2019). <https://doi.org/10.1145/3299771.3299779>

[3] Y. Sui, M. Zhou, M. Zhou, S. Han, D. Zhang, Table meets LLM: Can large language models understand structured table data? A benchmark and empirical study. *Trans. Struct. Data Intell.* **5**, 59–78 (2024). <https://doi.org/10.5555/tldi.2024.51059>

- [4] S. Maji, A. Appe, R. Bali, R. Ch., A. Chowdhury, V. Bhandaru, An interpretable deep learning system for automatically scoring request for proposals. *arXiv preprint arXiv:2008.02347* (2020). <https://doi.org/10.48550/arXiv.2008.02347>
- [5] A. MacLaughlin, S. Xu, D.A. Smith, Recovering lexically and semantically reused texts. In *Proceedings of the Tenth Joint Conference on Lexical and Computational Semantics (SEM 2021), Online, pp. 52–66(2021). <https://doi.org/10.18653/v1/2021.starsem-1.5>
- [6] S. Gullstrand, C. Lindén, Generative AI for sales: A study on the potential of retrieval-augmented generation for response automation in the request for proposal process in telecommunications. *Master Thesis*, KTH Royal Institute of Technology (2024). <http://kth.diva-portal.org/smash/record.jsf?pid=diva2:1884900>
- [7] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, H. Wang, Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997* (2023).
- [8] G. Convertino, V. Bellotti, N. Ikeya, O. Brdiczka, A. Hoshino, M. Sakaguchi, Y. Motohashi, H. Sakagamine, Opportunities for proactive support of reuse and organizational learning in ad hoc collaborative work activity: A field study of proposal writing. In *Proceedings of the International Conference on Collaboration Technologies and Systems (CTS 2011)*, pp. 143–150 (2011). <https://doi.org/10.1109/CTS.2011.5928676>
- [9] R. Zhang, Y.S. Wang, Y. Yang, Generation-driven contrastive self-training for zero-shot text classification with instruction-following LLM. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 659–673 (2024).
- [10] R. Jin, J. Du, W. Huang, W. Liu, J. Luan, B. Wang, D. Xiong, A comprehensive evaluation of quantization strategies for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 12186–12215 (2024).
- [11] J. Kim, M. Hur, M. Min, From RAG to QA-RAG: Integrating generative AI for pharmaceutical regulatory compliance process. In *Proceedings of the 40th ACM/SIGAPP Symposium on Applied Computing*, pp. 1293–1295 (2025).
- [12] S. Adhikari, Learning systems based automated proposal evaluation (2013).
- [13] J. Edwards, Knowledge management systems and business processes. *Int. J. Knowl. Syst. Sci.* **2** (2005).
- [14] W. Silva, S. Thelijjagoda, Managing the RFP process of government ICT projects from a discursive perspective. *Engineer: Journal of the Institution of Engineers, Sri Lanka* **57**, 81–92 (2024). <https://doi.org/10.4038/engineer.v57i4.7668>
- [15] D. Nguyen, S. Gupta, K. Do, T. Nguyen, S. Venkatesh, Generating realistic tabular data with large language models. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pp. 330–339 (2024).