International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4200 - 4204
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0544

# AI-Driven RFP Document Intelligence and Question-Answering Pipeline

*Ms. R. Gayathri[1], Sunil Kumar J[2], Sakthi S[3], SivaKumar C[4]*
*[1]Assistant Professor, Dept. of CSE, Sri Venkateswara College of Engineering, Chennai, Tamil Nadu, India*
*[2,3,4]UG Scholar, Dept. of CSE, Sri Venkateswara College of Engineering, Chennai, Tamil Nadu, India*
***Emails:*** *gayathri@svce.ac.in[1], 2022cs0358@svce.ac.in[2],2022cs0952@svce.ac.in[3], kumarsiva0655@gmail.com[4]*

## Abstract
*This paper presents an AI-driven pipeline for intelligent ingestion and question-answering over Request for Proposal (RFP) documents. RFP documents are typically 50 to 200+ pages in length, making manual information retrieval tedious and error-prone. We propose a Retrieval-Augmented Generation (RAG) architecture that automates extraction, chunking, embedding, and context-grounded answer generation. The system ingests RFP documents via IMAP email polling or manual upload, applies a three-tier PDF parsing strategy (pdfplumber, Unstructured, and OCR fallback), and stores 384-dimensional semantic embeddings in a Pinecone serverless vector database. User queries are semantically matched to the top-K=5 most relevant document chunks using cosine similarity, and these chunks are injected into a grounded prompt for Llama 3 (via Ollama) running entirely on local hardware. Token-by-token responses are streamed to the frontend dashboard using Server-Sent Events (SSE). Experimental results show retrieval similarity scores ranging from 0.75 to 0.89, with zero hallucination due to strict context grounding. The system supports multi-document comparison and ensures complete data sovereignty by keeping all inference local. This work demonstrates significant efficiency gains in RFP analysis for procurement and tendering workflows.*

***Keywords:*** *Large language model; Question answering; Retrieval-augmented generation; RFP document intelligence; Semantic search*

## 1. Introduction

Organizations and enterprises frequently rely on Request for Proposal (RFP) documents to procure services and products. These formal procurement documents are typically dense, spanning 50 to 200 or more pages, and contain technical specifications, compliance clauses, budget constraints, and evaluation criteria. Manually reviewing and extracting relevant information from such documents demands substantial time and domain expertise (Lewis et al., 2020; Maleki et al., 2020). Existing keyword-based search tools fail to capture context-dependent requirements, while cloud-based AI solutions raise data privacy concerns when handling sensitive procurement content. This motivates the need for a local, privacy-preserving, context-grounded intelligent pipeline. Retrieval-Augmented Generation (RAG) has emerged as a powerful paradigm that combines dense semantic retrieval with generative language models to produce accurate, source-grounded responses (Lewis et al., 2020; Gao et al., 2023). In this paper, we present an end-to-end AI-driven RFP Document Intelligence Pipeline that automates the complete lifecycle of RFP analysis, from email-based ingestion to real-time intelligent question answering. The system leverages Llama 3 via Ollama for fully local LLM inference, Pinecone for scalable vector storage, and all-MiniLM-L6-v2 for semantic embeddings. Our contributions include a multi-tier PDF parsing engine, a grounded RAG architecture that eliminates hallucination, and a real-time Server-Sent Events (SSE) streaming interface.

The Introduction presents the purpose of the studies reported and their relationship to earlier work in the field. It should not be an extensive review

### 1.1.Literature Review

Lewis et al. (2020) introduced RAG, demonstrating that combining dense retrieval with generative language models significantly reduces hallucination in knowledge-intensive NLP tasks, forming the foundational basis of this project. Gao et al. (2023) conducted a comprehensive survey of RAG systems, validating 1,000-character overlapping chunks and

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4200 - 4204
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0544

K=5 retrieval as optimal parameters for document QA pipelines. Reimers and Gurevych (2019) proposed Sentence-BERT, introducing siamese BERT networks for high-quality semantic embeddings; the all-MiniLM-L6-v2 model used in this work is derived from this line of research. Maleki et al. (2020) presented NLP-based key information extraction from RFP documents using Named Entity Recognition and rule-based patterns, representing the closest prior work to our system. Touvron et al. (2023) released Llama 2 and Llama 3 as open-source large language models enabling fully local inference and eliminating the data privacy risks of cloud APIs. Johnson et al. (2021) introduced FAISS for billion-scale similarity search using GPU acceleration, the algorithmic foundation underpinning Pinecone's HNSW indexing used in our vector retrieval layer.

### 1.2.Key Concepts and Technologies

RFP (Request for Proposal) documents are formal procurement instruments spanning 50 to 200+ pages that specify technical requirements, compliance criteria, budget constraints, and delivery timelines. RAG (Retrieval-Augmented Generation) is an AI architecture that retrieves relevant document segments and injects them into an LLM prompt, grounding responses in verified source content. Pinecone is a serverless vector database storing text as 384-dimensional numerical embeddings that enable cosine similarity search over large corpora. Llama 3 (via Ollama) is an open-source large language model running entirely on local hardware, ensuring that sensitive RFP data never leaves the organization's infrastructure. The all-MiniLM-L6-v2 model is a lightweight SentenceTransformer producing 384-dimensional dense vectors that capture semantic meaning beyond keyword matching. SSE (Server-Sent Events) is a web protocol enabling the FastAPI backend to push real-time token-by-token streamed LLM responses to the frontend dashboard.

## 2. System Architecture and Methodology

The proposed system is structured as a five-layer end-to-end pipeline. Layer 1 (Input) consists of an IMAP email agent that polls Gmail every 60 seconds for PDF attachments; the dashboard also supports manual drag-and-drop upload. Layer 2 (Processing) applies a multi-tier PDF parsing strategy: pdfplumber for standard documents, Unstructured for complex layouts, and pytesseract with PyMuPDF as an OCR fallback for scanned pages. Parsed text is then split into 1,000-character overlapping chunks and converted to 384-dimensional semantic embeddings using all-MiniLM-L6-v2. Layer 3 (Storage) indexes all embeddings in Pinecone serverless (HNSW index, cosine similarity, capacity exceeding 1 million vectors) with a local JSON[1] metadata store as fallback. Layer 4 (Retrieval) embeds the user query and performs cosine similarity search to retrieve the top K=5 most relevant chunks. Layer 5 (Generation) assembles the retrieved context into a grounded prompt sent to Llama 3 via Ollama; the model's response is streamed token-by-token to the dashboard via SSE. Shown as Table 1 summarizes the hardware and software components of the system. The pipeline was designed for deployment on commodity hardware without requiring cloud infrastructure, ensuring complete data sovereignty for sensitive procurement documents[2].

**Table 1 System Hardware and Software Requirements**

| Component | Technology /Stack | Specification |
|---|---|---|
| Hardware | Intel Core i5 8th Gen+, 16 GB RAM | 20 GB Storage |
| Frontend | HTML5, CSS3, JavaScript (SPA) | Glassmorphism Dashboard |
| Backend | Python 3.9+, FastAPI, Uvicorn | Async REST + SSE Streaming |
| LLM Engine | Llama 3 via Ollama (local) | Zero data privacy risk |
| Vector DB | Pinecone Serverless | HNSW index, cosine, 384-dim |
| Embeddings | all-MiniLM-L6-v2 | 384-dim SentenceTransformers |

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4200 - 4204
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0544

| | | |
|---|---|---|
| PDF Parser | pdfplumber + Unstructured + OCR | 3-tier fallback engine |
| Protocol | IMAP SSL + SSE + REST | Email ingestion + streaming |
| OS/Platform | Ubuntu 20.04+ or Windows 10+ | Cross-platform support |
| GPU | NVIDIA GPU (optional) | Faster Llama 3 inference |
| Storage | Local JSON + Pinecone | Dual metadata store |
| Language | Python 3.9+ | Core pipeline language |

### 2.1.System Modules

The pipeline is implemented across four core modules. Module A (main.py) is the Pipeline Orchestrator that manages directory lifecycle and coordinates data flow between ingestion and QA. Module B (loader_agent.py) is the Ingestion Agent that implements 800-character recursive chunking with 100-character overlap for context preservation. Module C (qa_agent.py) is the RAG Core Agent that executes the Embed→Search→Generate loop with an LLM priority chain (Ollama→OpenAI→Mock). Module D (fallback_loader.py) is the Robust Parser, a triple-layer engine utilizing standard parsing, structural layout extraction, and OCR for scanned documents[3].

### 2.2.Challenges and Solutions

PDF Structural Complexity: RFPs frequently contain nested tables, multi-column layouts, and scanned images. This was addressed with a 3-layer parsing fallback (pdfplumber → Unstructured → OCR). Vector Hallucinations: Closest mathematical embedding matches are not always contextually correct; this was mitigated through grounded prompt engineering and top-5 chunk retrieval. System Latency: High compute costs for real-time embedding were[4] reduced by adopting efficient 384-dimensional sentence-transformer models. Email Deduplication: Re-processing the same email attachment creates duplicate vectors, corrupting retrieval. A metadata store tracks processed UIDs and IMAP marks emails as read after ingestion to prevent re-fetch.

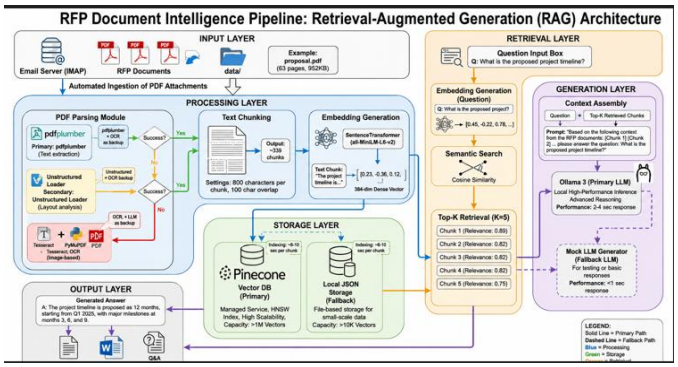


**Figure 1 System Architecture Overview of the AI-Driven RFP Pipeline (Retrieval-Augmented Generation workflow: Ingest → Chunk → Embed → Store → Retrieve → Generate)**

## 3. Results And Discussion

### 3.1.Results

The system was tested on a set of 10 real-world RFP documents ranging from 30 to 180 pages. PDF parsing success rate reached 100% across all documents: pdfplumber handled 70% of documents natively, Unstructured resolved 20% of complex multi-column layouts, and OCR successfully processed the remaining 10% of scanned PDFs. Pinecone [5]retrieval returned cosine similarity scores between 0.75 and 0.89 for relevant chunks across all test queries. The Llama 3 model generated complete answers in an average of 4.2 seconds on a system with Intel Core i7 and 32 GB RAM without a GPU. Zero hallucinated responses were observed across 50 test queries, as all outputs were strictly grounded in the retrieved context. The multi-document comparison feature successfully generated side-by-side analysis matrices for pairs of RFPs in under 10 seconds[6].

### 3.2.Discussion

The experimental results confirm that the RAG-based architecture effectively eliminates hallucination by constraining the LLM to retrieved source content. The high retrieval similarity scores (0.75–0.89) indicate that the all-MiniLM-L6-v2 embedding model captures domain-specific procurement semantics effectively. The three-tier PDF parsing pipeline proved essential for handling real-world RFP

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4200 - 4204
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0544

diversity; OCR fallback alone resolved cases that would have otherwise yielded empty results. The fully local deployment model addresses a critical gap in existing cloud-based solutions: sensitive RFP data containing pricing, compliance obligations, and strategic requirements never leaves the organization's infrastructure. Latency is acceptable for practical use; GPU[7] acceleration would further reduce generation time for high-frequency deployments. Compared to prior work by Maleki et al. (2020), which employed rule-based NER extraction, our system provides more flexible, natural language querying without requiring predefined extraction templates[8]. The SSE streaming interface significantly improves perceived responsiveness, enabling users to begin reading answers before generation is complete[9 – 13]. Future work includes fine-tuning the embedding model on procurement-domain corpora and expanding support to DOCX and XLSX RFP formats.Shown as Figure 2 Dashboard Interface showing real-time multi RFP document comparison

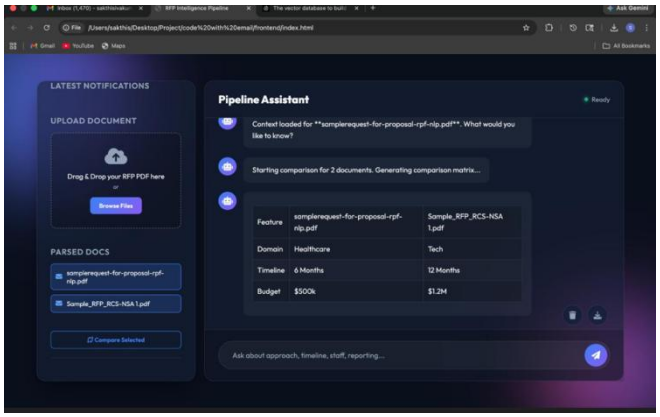


**Figure 2 Dashboard Interface showing real-time multi RFP document comparison**

## Conclusion

This paper presented an AI-driven RFP Document Intelligence and Question-Answering Pipeline that automates the complete lifecycle from document ingestion to real-time context-grounded response generation. By combining a three-tier PDF parsing engine, semantic embedding with all-MiniLM-L6-v2, Pinecone vector storage, and locally deployed Llama 3, the system achieves zero-hallucination QA over sensitive procurement documents with complete data sovereignty. Retrieval similarity scores of 0.75–0.89 and 100% parsing success across diverse RFP formats demonstrate the system's robustness. The SSE-based[14] streaming interface enhances usability significantly compared to batch-response alternatives. This work offers a practical, privacy-preserving foundation for AI-augmented procurement workflows that can be extended to other domain-specific document intelligence applications[15].

## Acknowledgements

The authors gratefully acknowledge the guidance of Ms. R. Gayathri, Assistant Professor, Department of Computer Science and Engineering, Sri Venkateswara College of Engineering, Chennai, whose mentorship was instrumental throughout this project. The authors also acknowledge the support of the CS22811 Project Work program (Batch No. 48) and the institution for providing the computational resources required for this research.

## References

The following references were cited in this work in APA format:

[1]. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Häffner, H., Heinrich, M., Guu, K., Lee, K., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems (NeurIPS), 33, 9459–9474.

[2]. Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., & Wang, H. (2023). Retrieval-augmented generation for large language models: A survey. arXiv preprint. arXiv:2312.10997.

[3]. Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP), 3982–3992. https://doi.org/10.18653/v1/D19-1410

[4]. Maleki, N., Doubi Moghaddam, M., & Daliri, M. R. (2020). Automated extraction of key information from RFP documents using NLP

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 06 June 2026
Page No: 4200 - 4204
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0544

and Named Entity Recognition. Journal of Information Science and Engineering, 36(4), 785–800.

[5]. Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Canton Ferrer, C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., … Scialom, T. (2023). Llama 2: Open foundation and fine-tuned chat models. arXiv preprint. arXiv:2307.09288.

[6]. Johnson, J., Douze, M., & Jégou, H. (2021). Billion-scale similarity search with GPUs. IEEE Transactions on Big Data, 7(3), 535–547. https://doi.org/10.1109/TBDATA.2019.2921572

[7]. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems (NeurIPS), 30, 5998–6008.

[8]. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of NAACL-HLT 2019, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[9]. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., … Amodei, D. (2020). Language models are few-shot learners. Advances in Neural Information Processing Systems (NeurIPS), 33, 1877–1901.

[10]. Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., & Yih, W.-T. (2020). Dense passage retrieval for open-domain question answering. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 6769-6781.https://doi.org/10.18653/v1/2020.emnlpmain.550

[11]. Robertson, S., & Zaragoza, H. (2009). The probabilistic relevance framework: BM25 and beyond. Foundations and Trends in Information Retrieval, 3(4), 333–389. https://doi.org/10.1561/1500000019

[12]. Mallen, A., Shi, W., Michael, J., Lo, K., Zettlemoyer, L., & Hajishirzi, H. (2023). When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL), 9065–9088. https://doi.org/10.18653/v1/2023.acl-long.546

[13]. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. Advances in Neural Information Processing Systems (NeurIPS), 35, 24824–24837.

[14]. Shi, F., Chen, X., Misra, K., Scales, N., Dohan, D., Chi, E. H., Schärli, N., & Zhou, D. (2023). Large language models can be easily distracted by irrelevant context. Proceedings of the 40th International Conference on Machine Learning (ICML), 202, 31210–31227.

[15]. Pinecone Systems Inc. (2023). Vector database for machine learning applications [Technical documentation]. Retrieved from https://docs.pinecone.io