

Design and Evaluation of a Retrieval-Augmented Generation System for Placement Data Querying in Academic Institutions

Amaan Kasu

Information Technology
Vidyalankar Institute of Technology
Mumbai, India
amaankasu08@gmail.com

Rashmi Sahu

Information Technology
Vidyalankar Institute of Technology
Mumbai, India
rashmi.sahu7039@gmail.com

Gaurav Khanna

Information Technology
Vidyalankar Institute of Technology
Mumbai, India
gauravkhanna702@gmail.com

Abstract — Acquisition placement cells have to deal with vast amounts of heterogeneous data, including spreadsheets, documents, and historical records, which cannot be queried efficiently by traditional search methods. The manual retrieval often causes delays, inconsistencies, and underutilization of data. This paper presents the design and evaluation of a RAG system tailored for querying placement data in academic institutions. The proposed system integrates semantic retrieval with large language models to support natural language querying over unstructured and semi-structured placement data. Rather than proposing a novel model architecture, this work focuses on applied system design, data ingestion optimization, secure deployment, and empirical evaluation under real-world constraints. Results demonstrate that the system significantly improves query accessibility and response relevance while maintaining stability and cost efficiency within an institutional environment.

Keywords — RAG, Document Intelligence, Semantic Retrieval, Vector Databases, Placement Analytics, Academic Information Systems, Natural Language Querying, Applied Machine Learning.

I. INTRODUCTION

The role of the academic placement cells includes the administration and dissemination of large amounts of student placement, company offers, student eligibility, company job descriptions, and statistical trends related to student placements. However, due to the growing nature of such information over the course of time, the data is accumulated across various forms such as Excel files, PDF files, MS Word files, and other emails containing attachments. Even though the information is of prime importance to the students, faculties, and the placement cells themselves, the access to the information still stands out to be one of the major issues. For example, to seek the company-wise hiring trends or the eligibility patterns, one needs to painstakingly go through the plethora of files related to the student placements.

Generally, traditional data retrieval approaches such as a keyword search system or a database system are inflexible instruments that are inefficient in dealing with such heterogeneous, semi-structured information, especially when dealing with keyword search limitations regarding information that might be expressed in different ways in different documents, as well as inflexibility in maintaining a database schema that could keep pace with the changing nature of placement information.

Recent advances in large language models (LLMs) offer a promising alternative by enabling natural language interaction

with data. However, direct use of LLMs is unsuitable for institutional environments, as these models lack access to private documents, may generate hallucinated responses, and cannot ensure grounding in authoritative data. To address these limitations, Retrieval-Augmented Generation (RAG) has emerged as an effective paradigm that combines semantic retrieval with controlled language generation.

In this work, we design and evaluate a RAG-based system specifically tailored for placement data querying in academic institutions. The system enables users to ask natural language questions and receive responses grounded in relevant placement documents. Instead of proposing a new model architecture, this paper focuses on applied system design, including ingestion of heterogeneous data, optimization of semantic indexing, secure deployment, and evaluation under real-world constraints. By addressing both technical and operational challenges, the proposed system demonstrates how retrieval-augmented systems can be practically adopted in academic administrative settings.

II. LITERATURE SURVEY

Retrieval-Augmented Generation was first proposed by Lewis et al. [1], showing how the use of a language model output grounded to relevant documents greatly improves the "factual correctness" for intensive knowledge-related tasks, and this is now common as the foundation for many retrieval approaches used in production enterprise/institutional settings.

In recent work, Karpukhin et al. introduced Dense Passage Retrieval (DPR), which demonstrated that embedding-based retrieval is superior to keyword search for answering semantic queries. This is directly applicable to placement datasets, as there might be variations for similar pieces of information.

Izacard and Grave reported an extension of the retrieval augmented QA method in the study titled "Increased Retrieval-Augmented Question Answering Using Multi-Passage Generation," emphasizing the need to choose a number of good context chunks before the actual generation.

Thakur et al. [4] have reported their work on the BEIR benchmarks, in which they demonstrated significant

differences in retrieval models across domains. The relevance of this is highlighted in this research using the academic placement data.

There has been work done on the deployment of such generative systems within the enterprise domain as investigated in the work of Vemulapalli [5], who described architectures for enterprise AI-powered chatbots. Perumallapalli [6] and Muntala [7] also presented on the deployment issues of a conversational AI within the enterprise domain.

Asfoura et al. in [8] suggested layered chatbot architecture models in the context of enterprise platforms, where layers of access, orchestration, and inference layering were highlighted, similar to the layered concept discussed in the current paper and presented in Fig. 1.

Techniques for scalable vector indexing were explored by Johnson et al. [11] and Malkov and Yashunin [12], thus facilitating the search operation for large sets of embeddings. Such techniques facilitate the possibility of a real-time semantic-based search operation for large placement archives.

In generative QA, hallucination was studied by Ji et al. in their paper [13], that successfully reduced the number of incorrect answers via retrieval-grounding generation and further validates our prompt system through retrieval.

Despite these advances, existing work largely focuses on model-level performance or benchmark evaluation. Limited research has addressed the end-to-end system design, deployment, and evaluation of RAG systems for academic institutional data, hence motivating the applied focus of this work.

From the above literature, it can be identified that though RAG architecture and semantic retrieval have garnered substantial interest in recent times, there seems to be a profound oversight when it comes to actual implementation and usage of RAG architecture in an institutional setting of data. All studies seem to be inclined toward constant improvements to individual models and performance benchmarking only.

This paper fills the research gap by presenting the design and evaluation of a proposed system for Retrieval-Augmented Generation particularly focusing on a system designed to aid the querying of placement information within an educational institution. The novelty is not necessarily within the design of a novel architecture, as the novelty of the presented research lies within the demonstration of the practical deployment, optimization, and evaluation of an overall RAG system.

III. SYSTEM ARCHITECTURE

Overall architecture for the proposed intelligent system for placements is presented in Fig. 1. The intelligent system for placements is presented in a layered modular architecture to enable access security, dissimilar data retrieval, and response

generation considering heterogeneities present in the data for placements. Each separate layer is independently deployable, considering the constraints imposed by institutions for access to a specific network, cost factor, and data privateness.

A. Overview of Fig. 1

From Fig. 1 above, the major layers of the system comprise the user access layer, the internal orchestration layer, the inference layer, the storage layer, and the infrastructure and security services layer. As such, it becomes apparent how the system design allows it to remain an extensible, maintainable, and robust system as the data and query increase.

The architecture here follows a controlled access pattern where all requests by the users go through an internal media with end-to-end encryption and then interact with the services and components involved without leaking any data outside.



Fig. 1 Layered system architecture of the proposed *PlacifyAI* retrieval-augmented generation system, showing secure user access, internal orchestration, inference services, storage components, and infrastructure-level security controls.

B. User Access Layer

The User Access Layer serves as a secure access point, which, in this case, would comprise end users such as students, faculty, as well as placement officers. Users are able to interact with the system via a web-based front end, which has been implemented as HTTPS/TLS traffic. On the other hand, incoming connections are facilitated by a secure channel that eliminates direct exposure to the internal network.

This layer is the sole implementation of request forwarding and authentication. This ensures the client-side implementation is minimal.

C. Internal Orchestration Layer

The layer in charge of the internal orchestration of the entire system is the Internal Orchestration Layer. It consists of an application server that processes the user queries and a task worker engaged in the execution of document ingestion and generation of document embeddings in the system in a background manner.

The orchestration level of the system also facilitates the communication among the components of the storage, generation, and the queries.

D. Inference Layer

The Inference Layer comprises the components for semantic indexing and response generation. It includes the service for embedding generation during the ingestion phase and the one for performing the language model inference during the query phase. This provides the ability to scale the indexing and query workload independently. It also allows routing requests to multiple models for satisfying the cost, latency, or availability-based constraints.

E. Storage Layer

The storage layer manages structured as well as unstructured data storage. It supports a relational database to store user searches' metadata, user logs' metadata, and ingestion state. An in-memory store can store caches and facilitate job coordination to enhance system responsiveness. Lastly, embeddings are stored in a dedicated search engine meant for efficient support of vector search in documents' datasets. Such a multi-storage solution suits different kinds of datasets to different kinds of storage solutions most appropriate to them.

F. Infrastructure, Security and External Services

The system is deployed into a secure Linux-based virtual machine infrastructure with contained services to guarantee portability and reproducibility. For network-level security, there is enforcement of identity-based access, regulated outbound connection, and reverse routing proxies. For the external language model providers, access is done only through regulated network paths to prevent direct exposure to internal data.

G. Architectural Design Considerations

- In addition to the physical structure, a range of architectural principles are also employed, including:
- Security by Isolation: Components are not directly exposed.
- Scalability through modularity: All the stacked layers can scale on their own.
- Reliability through orchestration: Errors during ingestion or indexing do not propagate through user queries.
- Cost Awareness: "Model usage and retrieval depth are dynamically controlled."
- Institutional suitability: The architecture is optimized for an environment with limited resources and is suitable for academic institutions with stringent rules in place.

IV. METHODOLOGY AND SYSTEM DESIGN

Overall, the proposed system is based on a modular approach that utilizes the Retrieval Augmented Generation (RAG) pipeline, which is best suited to cater to large volumes of placement data that are diverse while being confronted with real-life institutional needs.

While the methodology does not focus on any major innovations with regard to the model itself, the approach does mainly concentrate on the ingestion,

A. Data Sources and Preprocessing

Placement data is collected from a variety of unstructured and semi-structured data sources like Excel sheets, PDF files, Word files, and placement summaries. These data have varying levels of structure, large varying sizes, and quality levels between them. A preprocessing paradigm is adopted to normalize the data to varying levels by extracting the information, removing redundant data, and dealing with large and irregular table-based data common in placement data sheets. The data is segmented in semantically consistent chunks to enable efficient information retrieval.

B. Embedding and Vector Indexing

Each chunk is embedded as a dense vector through the transformation model embedding method. Such vectors represent semantic similarity instead of string matching based on words. They are used to implement natural language search by the documents. They are also stored in a database that supports efficient search for the nearest neighbors to retrieve the desired information in a short time as the corpus of documents grows. Additional information, such as the publication date, company name, or document type, is also embedded in the vectors.

C. Retrieval and Answer Generation

When a user poses a query, the query is embedded and matched with the vectors generated while indexing documents. Relevant portions are retrieved in the form of the top k relevant chunks, which are then used as the input for the large language model. This minimizes the risk of hallucination found with the answers generated as the answers are strictly based on the content retrieved.

D. System Deployment and Security

The system is hosted on a secure cloud-based virtual machine; access is granted through a containerized service-based system, which is only accessible through internal endpoints and monitored for unauthorized data egress. It is a region-aware and budget-friendly system to enable a consistent and stable performance within a specified budget. It is web-based to enable authorized users to access and query the system without access to actual data for enhanced data privateness.

E. Design Rationale

There is an emphasis on the robustness, scalability, and maintainability of the methodology. Unlike other

methodologies where the novelty of the algorithm is sought to be highlighted, the novelty of the RAG principles is showcased through the adaptability, optimization, and utilization of these principles in the arena of academic administration.

V. IMPLEMENTATION DETAILS

This section presents the overall implementation of the proposed approach on the problem of data querying on placement data, termed Retrieval-Augmented Generation (RAG). Figure 2 presents the overall data processing pipeline with regard to the proposed approach on data querying on placement data, emphasizing reliability, as well as relevance rather than architectural novelty.

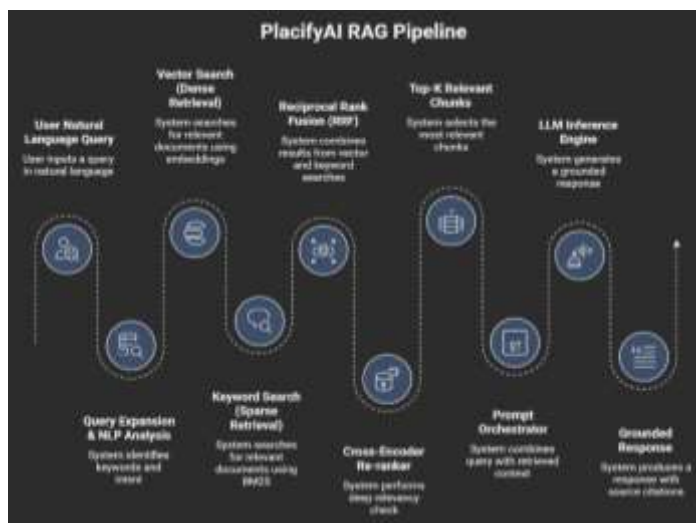


Fig. 2 End-to-end retrieval-augmented generation pipeline of the PlacifyAI system, illustrating query analysis, hybrid retrieval, ranking, prompt orchestration, and grounded response generation.

A. Query Processing and Understanding

As depicted in Fig. 2 below, a user starts a conversation by making a natural language query via a user-friendly interface on the web. The process of processing the user's query then starts from the module labeled "Query Expansion and NLP Analysis," which helps in determining what a user wants as well as other considerations from a contextual point of view. A user's query might vary in terms of wording despite being a similar query relating to a similar piece of information in terms of placements such as eligibility criteria, company-specific info, or historical placements.

B. Hybrid Retrieval Strategy

With the need for a perfect balance in terms of semantic as well as lexical information, the mixed retrieval model is used.

1. Dense Retrieval (Vector Search):

Further, the processed query is transformed into an embedding that is then matched with embeddings stored in the vector database. This process retrieves semantically relevant document sections that are not even matched on the level of keywords.

2. Sparse Retrieval (Keyword Search):

In parallel, a keyword-based search is performed using standard information retrieval methodologies such as BM25.

Such an approach proves to be powerful in dealing with structural information like company details, dates, or numerical placing statistics. Both of these retrieval paths result are then combined using a Reciprocal Rank Fusion-based model called RRF, providing a robust result set based on semantic similarity and keyword search algorithms.

C. Re-ranking and Context Selection

These merged candidate documents are fed into a cross-encoder re-ranking module where a deeper level of relevance is checked with reference to the query and each chunk. This enhances the overall answer quality by removing loosely related results and favoring directly useful context. From the results returned by the re-ranking module, the system selects the top k most relevant chunks of context, which are fed into the language model as grounding.

D. Prompt Orchestration and Generation

The Prompt Orchestrator processes the user query in conjunction with the contextual chunks selected in the previous step to formulate the final structured prompt, which is then fed into the LLM Inference Engine to obtain a response based only on the downloaded placement data. This minimizes the chance of "hallucination."

E. Grounded Response Delivery

The final output, as seen at the very right end of Fig. 2, is a grounded response with or without source document references. This response is returned to the user interface in near real-time latency, not exposing any internal storage or raw documents, thus ensuring data privacy.

F. Summary of Overall Project Flow

The PlacifyAI system follows a clear operational flow:

1. The operation sequence of the PlacifyAI system goes through clear stages:
2. The data of placement, therefore, is ingested, cleaned, chunked, embedded, and indexed offline.
3. Users' natural language queries are taken as input through a secure interface.
4. Queries are expanded and passed through the hybrid retrieval.
5. Retrieved results are fused, re-ranked, and filtered.
6. Language Model generates a grounded response using verified context.
7. This means providing users with accurate, traceable answers in real time.

This pipeline allows for the sound operation of the system on heterogeneous, historically accumulated placement data while maintaining performance, security, and institutional suitability.

VI. RESULTS AND DISCUSSION

A. Comparative Analysis with Existing Systems

Feature	Traditional Manual Methods	Generic Document Search Systems	PlacifyAI (Proposed Systems)
Query Interface	Keyword-based or manual browsing	Keyword or limited semantic search	Natural language querying
Data Format Support	Limited, manual handling	Partial (mostly text documents)	Excel, PDF, Word, mixed formats
Retrieval Accuracy	Low, user-dependent	Moderate	High (semantic retrieval)
Response Generation	Not available	Not available	Context-aware, grounded responses
Time Efficiency	Low	Moderate	High
Domain Customization	None	Generic	Placement-specific optimization
Deployment Suitability for Colleges	High effort, manual	Often enterprise-focused	Designed for academic institutions
Data Privacy Control	File-level access only	Cloud-dependent	Controlled, institution-ready deployment
Decision-Making Support	Minimal	Limited	Strong, actionable insights

A test bed was designed to assess the proposed plan of a retrieval-augmented generative system based on the significant impact that such a system could present and the efficiency attained within an educational institution. Fig. 3 presents a comparative visualization of the proposed plan and other approaches adopted to retrieve placement data.

Traditionally, placement officers as well as academics have to rely on manual scanning and searching through numerous Excel sheets, PDF format reports, and word documents to obtain the placement data as presented in Fig. 3. This entire process is found to be highly cumbersome, creating issues of inaccuracies due to data duplication, outdated data stored in the documents created by placement officers, as well as manual interpretation errors by individuals involved in the process.

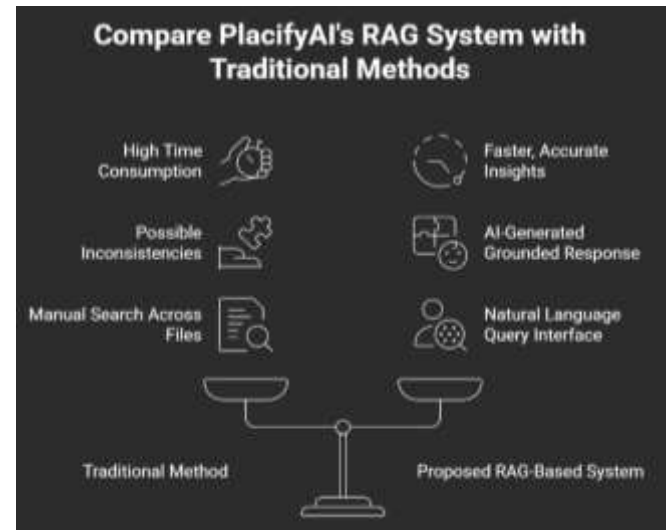


Fig. 3

On the contrary, the proposed system allows users to query placement information by natural languages through a uniform interface. It exploits the benefit of semantic retrieval to obtain relevant information from all available sources connected to the system, irrespective of their data format and structure. Responses provided by the system have founded their origin in the extracted placement data from sources, increasing both precision and trustworthiness as well as efficiency in retrieving information such as company-wise hiring trends, eligibility criteria, and placement statistics for particular years by users.

The impact of this system is particularly relevant in academic institutions where technical infrastructure may not be available in abundance. By emphasizing application-oriented system design, deployment, and usage costs, this approach appears to be particularly useful in academic colleges where maintaining a sophisticated analytics infrastructure may not be feasible. The system minimizes reliance on technical expertise in handling complex infrastructure, thereby optimizing planning in various academic drives and providing essential guidelines to students in general academic reporting.

Additionally, the system, unlike other general enterprise document assistants, is designed to take into account special data patterns specific to placement, such as large tabular records provided in the data set, as well as historical document accumulation.

The results show that the proposed RAG-based intelligent system can improve efficiency and decrease manual intervention, and can enhance the practical applicability of placement data maintained by institutions. The findings indicate significant opportunities and possibilities for the adoption of the proposed intelligent system in various academic institutions without altering existing infrastructures.

VII. CONCLUSION AND FUTURE WORK

SmartSpace, a browser-native, totally open-source platform that significantly democratizes professional-grade 3D interior design, was introduced in this paper. SmartSpace removes the long-standing gap between conceptual planning and spatial visualization by combining a topologically sound half-edge data model with real-time bidirectional synchronization between an intuitive 2D floor plan editor and a high-fidelity 3D renderer. Without the need for installation, plugins, or costly licenses, the system enables precise wall thickness and

apertures, achieves interactive frame rates (>55 FPS) on commodity hardware, and offers constraint-aware item placement. Its practicality and performance advantages over current web-based alternatives are confirmed by empirical results and real-world deployments in e-commerce and real-estate staging.

The following important extensions will be the focus of future work:

- Conflict-free replicated data types (CRDTs) allow for real-time collaborative editing to facilitate concurrent multi-user design sessions.
- On gadgets like Meta Quest and Apple Vision Pro, native WebXR integration allows for AR furniture placement and immersive VR walkthroughs.
- NURBS or Bézier edge representations are used to support curved and sloped walls in order to increase architectural expressiveness.
- Integration of style transfer and layout optimization powered by machine learning for automated design support.
- Improved compatibility for professional downstream workflows through standard export formats (IFC, glTF with BIM information).

SmartSpace demonstrates that advanced spatial computing can—and should—be publicly available in all browsers, reducing obstacles to excellent interior design for academics, professionals, and homeowners alike.

You may find the full source code at <https://github.com/amaankasu/PlacifyAI-Final>

REFERENCES

[1] P. Lewis et al., “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” NeurIPS, 2020.

[2] V. Karpukhin et al., “Dense Passage Retrieval for Open-Domain Question Answering,” EMNLP, 2020.

[3] G. Izacard and É. Grave, “Leveraging Passage Retrieval with Generative Models for Open-Domain QA,” EACL, 2021.

[4] N. Thakur et al., “BEIR: A Heterogeneous Benchmark for IR Models,” NeurIPS, 2021.

[5] V. K. C. Vemulapalli, “Enterprise Generative AI Chatbot Architecture,” 2023.

[6] R. Perumallapalli, “Conversational AI for Customer Support,” 2022.

[7] P. S. R. P. Muntala, “AI-Powered Chatbots in Enterprise Applications,” 2021.

[8] E. Asfoura et al., “Chatbot Conversational Systems for Enterprise Platforms,” Procedia CS, 2020.

[9] D. Völker and M. Oberhauser, “Agentic AI in Enterprise Systems,” IEEE Access, 2023.

[10] S. Hosseini and H. Seilani, “Agentic AI: A Systematic Review,” ACM CSUR, 2023.

[11] J. Johnson et al., “Billion-Scale Similarity Search with GPUs,” IEEE Big Data, 2019.

[12] Y. Malkov and D. Yashunin, “HNSW: Efficient ANN Search,” IEEE TPAMI, 2018.

[13] Z. Ji et al., “Hallucination in Natural Language Generation,” EMNLP Findings, 2023.