SENIOR PROJECT II (SPRING 2025)

FINAL REPORT

INSTRUCTOR: DR. THABET KACEM

---

# RETRIEVAL AUGMENTED GENERATION FOR RESUME ANALYSIS

---

Hugh Smith (N00296454, hubert.smith@udc.edu)

Department of Computer Science and Information Technology

University of the District of Columbia

4200 Connecticut Avenue NW

Washington, DC 20008

# Contents

# 1 PROJECT SUMMARY

The job market has become rough for applicants and companies looking to hire. Recruiters must sift through hundreds, if not thousands, of resumes to find a good match for a role, often relying on outdated keyword-based systems that do not capture the deeper nuances of candidate profiles. To address this challenge, we have designed an innovative application that leverages advanced Artificial Intelligence (AI) techniques, specifically Retrieval-Augmented Generation (RAG), to transform how resumes are searched, analyzed, and matched.

# 2 INTRODUCTION

This project addresses modern recruitment's increasing complexity, where job seekers and hiring companies face significant challenges. The recruitment process often involves sifting through numerous resumes, creating a substantial bottleneck for hiring managers attempting to identify suitable candidates efficiently. This problem is further exacerbated by the limitations of traditional keyword-based search systems, which struggle to capture the nuances of natural language and frequently overlook qualified applicants whose resumes may not contain specific keywords despite possessing relevant skills and experience. These inefficiencies in the hiring process lead to missed opportunities for both candidates and employers, preventing organizational growth and individual career progression. Crucially, handling sensitive personal data within resumes raises significant privacy concerns that this project directly addresses. This project aims to design and implement a Retrieval Aggregated Generation (RAG) system to revolutionize resume analysis and matching while prioritizing candidate privacy. This system will leverage advanced AI techniques to go beyond simple keyword matching, enabling a more nuanced understanding of candidate experience and qualifications through natural language understanding and contextual analysis. A key focus of this project is to implement a privacy-preserving information retrieval mechanism. Using advanced techniques, we aim to perform semantic searches and matching without directly exposing sensitive personal information contained within resumes. This approach allows for efficient and accurate matching while minimizing the risk of data breaches and ensuring compliance with privacy regulations. This project will explore the application of RAG architecture, coupled with this privacy-preserving retrieval method, to effectively retrieve relevant information from a corpus of resumes and generate insightful comparisons against

given job descriptions. This project will focus on the design, implementation, and evaluation of this RAG-based solution, aiming to demonstrate its potential to significantly improve the efficiency and effectiveness of the resume screening process within HR departments, ultimately leading to better hiring outcomes for both employers and job seekers while upholding stringent privacy standards.

# 3 PROJECT TIMELINE AND TASK OUTLINE

| **Project Timeline** | | | |
|---|---|---|---|
| **Task** | **Start Date** | **End Date** | **Duration** |
| *Phase 1: Research and Exploration* | | | |
| RAG Research | Sep 2 | Sep 9 | 1 Week |
| Model Exploration | Sep 9 | Sep 16 | 1 Week |
| Code Analysis | Sep 16 | Sep 27 | 1.5 Weeks |
| SLM Research | Sep 30 | Oct 4 | 1 Week |
| PHI-3 Testing | Oct 7 | Oct 11 | 1 Week |
| PDF Test Case | Oct 14 | Oct 18 | 1 Week |
| AI Expert Consultation | Oct 29 | Oct 29 | 1 Day |
| *Phase 2: Implementation and Design* | | | |
| Dataset Acquisition | Oct 30 | Nov 1 | 1 Week |
| RAG Architecture | Nov 1 | Nov 5 | 1 Week |
| GUI Exploration | Jan 15 | Jan 19 | 1 Week |
| Code Adaptation | Feb 1 | Feb 14 | 2 Weeks |
| Code Refactoring | Feb 17 | Feb 29 | 2 Weeks |
| Chatbot Development | Mar 1 | Mar 21 | 3 Weeks |
| Authentication Setup | Mar 22 | Apr 5 | 2 Weeks |
| *Phase 3: Finalization and Testing* | | | |
| End-to-End Testing | Apr 8 | Apr 15 | 1 Week |
| MVP Development | Apr 16 | Apr 26 | 1.5 Weeks |

**Table 1:** Project Timeline Overview

# 4 PREVIOUS WORK

The challenge of efficiently and effectively matching resumes to job descriptions has been a subject of ongoing research and development within the Natural Language Processing (NLP) and Information Retrieval (IR) fields. [1] Traditional approaches have often relied on keyword-based matching, which, as discussed earlier, needs to be improved in capturing the semantic nuances of both resumes and job descriptions. These systems often fail to recognize synonyms, related skills, or the context in which keywords appear, leading to false positives and false negatives in candidate selection.[2] Recent efforts have explored more sophisticated NLP techniques, including Semantic Analysis, Named Entity Recognition, and Topic Modeling. These methods aim to extract deeper meaning from textual data, allowing for more accurate matching based on skills, experience, and qualifications. For example, Word2Vec and GloVe have been employed to create word embeddings, representing words as vectors in a high-dimensional space where semantically similar words are located closer together. This allows systems to identify matches even when different wording is used. [3] The advent of transformer-based models, such as Bidirectional Encoder Representations from Transformers (BERT) and its variants, has further advanced the field. These models excel at understanding context and capturing complex relationships within text, significantly improving various NLP tasks, including text classification, question answering, and semantic similarity. Applying these models to resume matching allows for a more comprehensive understanding of candidate profiles. [4] However, privacy concerns have become paramount with the increasing use of sensitive personal information in resume analysis. Traditional methods of storing and processing resume data often involve storing the raw text, which raises risks of data breaches and misuse. This is where the concept of privacy-preserving information retrieval becomes crucial. Recent research has explored using vector databases and embedding models to address these privacy concerns. [5] By converting textual data into vector embeddings and storing only these embeddings in the database, the system can perform semantic searches without directly accessing or storing the raw, sensitive data. This approach significantly improves data protection and aligns with growing privacy regulations. [6] Techniques like federated learning and differential privacy have also been explored with vector databases to enhance privacy by allowing models to learn from decentralized data without directly sharing sensitive information. This project builds upon these advancements by explicitly focusing on applying a RAG architecture combined with a privacy-preserving vector database approach, leveraging the benefits of contextual understanding while mitigating privacy risks. Using a vector

database like Pinecone represents a specific implementation of this privacy-preserving strategy, allowing for efficient similarity searches without compromising sensitive data, a topic we will explore further in subsequent sections.

# 5 PROJECT APPROACH

This project systematically designs and implements a privacy-preserving RAG system for resume analysis. The core methodology can be broken down into the following key components:

1. Understanding and Implementing RAG: The foundation of this project lies in a thorough understanding of the Retrieval-Augmented Generation paradigm. RAG combines the strengths of information retrieval and generative language models. The retrieval component is responsible for fetching relevant context from a knowledge base (in this case, a collection of resumes), while the generation component uses this retrieved-context to generate a response or perform a specific task, such as comparing a resume against a preprogrammed job description. This project will explore different retrieval strategies and generation techniques to optimize the system's performance.

2. Dataset Acquisition and Preparation: A crucial step in any machine learning project is the acquisition and preparation of a suitable dataset. For this project, a comprehensive dataset of over 3000 PDF resumes sourced from Kaggle was selected. This dataset encompasses various career fields and resume formats, providing a realistic and challenging testbed for the RAG system. The PDF format presents an additional challenge, requiring robust PDF parsing and text extraction techniques to convert the resume content into a usable format for the model. This preprocessing step is essential for ensuring the quality and consistency of the data used for retrieval and generation.

3. User Interface Development: To ensure the RAG system is accessible and intuitive, a web-based interface was considered essential. Initially, frameworks such as Gradio and Streamlit were explored due to their low-code interfaces and rapid prototyping capabilities. While Streamlit was briefly used in the early stages for proof-of-concept development, we ultimately transitioned to a custom-built solution using TypeScript, React, and Shadcn UI. This shift provided greater flexibility, scalability, and design control, allowing for more

advanced user interactions, seamless authentication (Clerk) integration, and an overall production-ready architecture.

4. Security and Authentication: Given the sensitive nature of the data involved, implementing robust security measures is paramount. The system employs Clerk for user authentication, ensuring that only authorized users can access the application and its features. This authentication layer is critical for protecting sensitive personal information contained within resumes and maintaining compliance with privacy regulations. The integration of Clerk provides a secure and user-friendly authentication experience, allowing users to log in seamlessly while ensuring their data remains protected.

5. Privacy and Ethical Considerations: As mentioned above, a central focus of this project is addressing the privacy implications of handling sensitive personal information contained within resumes. Traditional methods of storing and processing resume data raise significant privacy concerns. This project aims to mitigate these risks by employing additional privacy-preserving techniques, primarily through vector embeddings and a specialized vector database. The system can perform semantic searches and comparisons without directly accessing or storing the raw, sensitive data by converting resume text into vector representations. This approach minimizes the risk of data breaches and aligns with growing privacy regulations. Furthermore, ethical considerations surrounding bias in AI models and fairness in hiring practices will be carefully considered throughout the project lifecycle. This includes evaluating the model's performance across different demographic groups and implementing strategies to mitigate potential biases.

This multi-faceted approach, encompassing RAG implementation, comprehensive dataset utilization, user-friendly interface development, and a strong emphasis on privacy and ethical considerations, forms the core of this project.

## 5.1 Retrieval-Augmented Generation Overview

"Retrieval Augmented Generation (RAG) is a framework that addresses this limitation by combining retrieval-based approaches with generative models. It retrieves relevant data from external sources in real-time and uses this data to generate more accurate and contextually relevant responses." [7] RAG offers a compelling approach to the complex resume analysis and matching

task. Combining the strengths of information retrieval and generative language models, RAG addresses the limitations of traditional keyword-based systems and provides a more nuanced and context-aware understanding of candidate profiles. This section outlines the RAG paradigm and its specific application within this project, focusing on how it enhances resume analysis. [7] The core idea behind RAG is to enhance the generation process with relevant information retrieved from an external knowledge source. In the context of resume analysis, this knowledge source is a collection of resumes and job descriptions. The RAG process can be broken down as follows:

1. **Ouery/input:** The process begins with a user query or input. For this project, the input is a resume or set of resumes that needs to be analyzed against a job description. The input can be in the form of a natural language question or a specific task, such as comparing a resume to a job description.

2. **Retrieval:** A retrieval component searches a large corpus of documents (our knowledge base) for information relevant to the input query. This project's knowledge base consists of a dataset of over 3000 PDF resumes sourced from Kaggle. The retrieval process aims to identify resumes (or relevant sections within resumes) that contain skills, experience, keywords, or other information pertinent to the job description (or the provided resume). For example, if the job description requires "project management experience," the retrieval component will search for resumes that explicitly mention "project management" or related terms like "project leadership," "team coordination," or specific project management methodologies.

3. **Augmentation:** The retrieved information is then combined with the original input query. This augmented input provides the generative model with the necessary context to perform the analysis. For example, if a job description requires proficiency in Python and the retrieval step finds several resumes mentioning Python experience, this information is combined with the job description and passed to the generative model.

4. **Generation:** Once the relevant context is retrieved, the generative language model uses this information to generate responses or perform specific tasks. In this project, the generation component will be responsible for comparing resumes against job descriptions, providing insights into how well a candidate's qualifications align with the requirements of a given role.

5. **Integration:** The integration of retrieval and generation is a key aspect of RAG. By combining these two components, the system can leverage the strengths of both approaches. The retrieval component ensures that the generative model has access to relevant and up-to-date information, while the generation component provides a flexible and context-aware response.

6. **Privacy Preservation:** A critical consideration in this project is the privacy of sensitive personal information contained within resumes. The RAG system will employ privacy-preserving techniques, such as vector embeddings and a specialized vector database, to ensure that sensitive data is not directly accessed or stored. This approach minimizes the risk of data breaches and aligns with growing privacy regulations.

The RAG architecture allows for a more sophisticated and context-aware analysis of resumes, enabling the system to understand the nuances of candidate profiles and job requirements. By leveraging the strengths of both retrieval and generation, RAG can provide more accurate and relevant insights into candidate qualifications, ultimately improving the efficiency and effectiveness of the resume screening process. A good RAG system contains two main components: a retriever and a generator. The retriever is responsible for fetching relevant documents from a large corpus, while the generator uses these documents to produce coherent and contextually relevant responses. This architecture allows the system to leverage the strengths of both retrieval-based and generative approaches, resulting in improved performance across various tasks. [7]

## 5.2 RAG Workflow

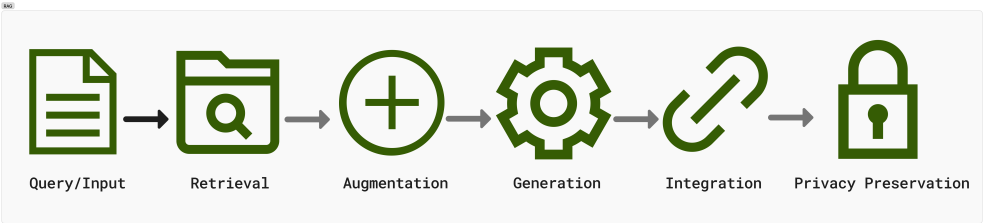


**Figure 1:** Shows the basic idea behind Retrieval Augmented Generation (RAG) and how it works.

By incorporating a retrieval step, RAG offers several key advantage for this project:

1. **Improved Accuracy and Relecance:** By grounding the analysis in a wide array of resumes, RAG can provide more accurate and relevant insights into candidate qualifications. This is particularly important in the context of resume analysis, where the nuances of language and context play a significant role in understanding a candidate's experience and skills.

2. **Dynamic Knowledge Base:** The retrieval component enables the system to access a dynamic knowledge base of resumes, ensuring that the analysis is based on up-to-date information.

3. **Privacy Preservation:** By employing privacy-preserving techniques, RAG can perform semantic searches and comparisons without directly accessing or storing sensitive personal information.

4. **Flexibility and Adaptability:** The RAG architecture allows for flexibility in adapting to different tasks and domains, making it suitable for various recruitment scenarios.

RAG provides a powerful and adaptable framework for this resume analysis and matching project, offering a significant improvement over traditional methods by leveraging the power of both retrieval and generation.

# 6 DATASET ACQUISITION AND PREPARATION

The effectiveness of a RAG system relies heavily on the quality and comprehensiveness of its knowledge base. In this project, the dataset of over 3000 resumes serves as this crucial knowledgebase, providing the foundation for robust and accurate resume analysis. The dataset's dual format, encompassing both string and PDF representations of resumes, is particularly beneficial. The string format allows for efficient indexing and retrieval of textual information, which is vital for the RAG pipeline. The inclusion of PDFs reflects real-world scenarios and necessitates implementing robust PDF parsing techniques, ensuring the system can handle the diverse formats encountered in recruitment workflows. The dataset was sourced from Kaggle, a well-known platform for data science and machine learning resources. Crucially, the dataset exhibits significant diversity across 24 distinct job categories. This breadth of categories, from common fields like Information Technology, Engineering, and Healthcare to more specialized areas such as Digital Media, Fitness, and Public Relations, is essential for the project's goals. In a

RAG system, the retrieval component searches this knowledge base to find relevant context for the generation component. The diversity of the dataset ensures that the system can effectively retrieve relevant information for a wide range of job descriptions and candidate profiles. For instance, if a user provides a job description for a "Software Engineer," the RAG system will search the resume database, retrieving resumes categorized as "Information-Technology" or "Engineering," as well as resumes containing relevant keywords like "programming," "software development," or specific programming languages. This retrieved information then augments the input to the generation component, allowing it to make informed comparisons and assessments. Therefore, the dataset's diversity is not primarily about training a statistical model to generalize from examples, as in traditional machine learning. Instead, it's about providing a rich and representative collection of resumes that the RAG system can effectively search and utilize to provide contextually relevant and accurate analysis for various job descriptions and candidate profiles. This robust and diverse knowledge base empowers the RAG system to perform effective resume analysis and matching. [8]

# 7 User Interface Development

The user interface (UI) of the RAG system is a critical component that directly impacts user experience and interaction with the application. Initially, low-code frameworks like Gradio and Streamlit were considered for rapid prototyping. However, these frameworks were ultimately deemed insufficient for the project's long-term goals, particularly regarding customization and scalability.

# 8 User Interface Framework Evaluation

A crucial aspect of this project was to provide a user-friendly and intuitive interface that would allow seamless interaction with the Retrieval-Augmented Generation (RAG) system. Early in the project, Gradio and Streamlit were evaluated as potential frameworks for building this interface. This section presents an overview of each framework, weighs their respective advantages and disadvantages in the context of the project, and explains the decision to adopt Streamlit.

## 8.1 Gradio

Gradio is a Python library designed for rapidly creating interactive interfaces for machine learning models. It excels at building simple demos and facilitating model sharing.

- **Advantages:**
  - **Ease of Use:** Gradio is extremely easy to set up, requiring minimal code to create basic interfaces.
  - **Built-in Components:** It offers a variety of built-in input and output components, enabling rapid prototyping without extensive customization.
  - **Sharing Capabilities:** Gradio allows for immediate sharing of interfaces via publicly accessible links, which facilitates collaboration and feedback.
- **Disadvantages:**
  - Limited customization options for user interface design and layout.
  - Less suitable for complex applications requiring advanced interactivity and dynamic layouts.

## 8.2 Streamlit

Streamlit is an open-source Python library that enables the creation of visually appealing and interactive web applications, particularly for machine learning and data science projects.

- **Advantages:**
  - **Flexibility and Customization:** Streamlit provides extensive control over the user interface, allowing developers to build complex and highly customized applications.
  - **Data Visualization:** It offers strong support for data visualization libraries, making it ideal for displaying the results and insights generated by the RAG system.
  - **Pythonic Development Workflow:** Streamlit adheres closely to Python programming paradigms, enabling efficient development by data scientists and machine learning engineers.

  - **Caching Mechanism:** Streamlit includes a built-in caching system that enhances application performance, especially for computationally intensive operations.

- **Disadvantages:**

  - Steeper learning curve compared to Gradio, particularly for users unfamiliar with web development principles.
  - More setup and configuration required to create a fully functional and polished application.

## 8.3 Comparison Summary

**Table 2:** Comparison of Gradio and Streamlit Frameworks

| **Feature** | **Gradio** | **Streamlit** |
|---|---|---|
| Ease of Use | Very simple setup, ideal for quick demos | Requires more configuration for complex apps |
| Customization | Limited UI customization | Highly customizable layouts and interactivity |
| Sharing | Easy link sharing built-in | Requires separate hosting for public sharing |
| Data Visualization | Basic support | Strong integration with visualization libraries |
| Performance | Lightweight for simple models | Better suited for complex and data-driven apps |

## 8.4 Final Decision

Ultimately, the decision was made to not work with either Gradio or Streamlit. The project pivoted towards a custom-built solution using TypeScript, React, and Shadcn UI. This decision was driven by the need for greater flexibility, scalability, and design control. The custom-built solution allows for more advanced user interactions, seamless integration with authentication (Clerk), and an overall production-ready architecture. Initially, Gradio was used briefly to provide a proof-of-concept for the RAG system. However, as the project evolved, it became clear that a more robust and customizable solution was needed to meet the project's long-term goals.

This approach offers several advantages:

- **Customization:** Full control over UI design and functionality enables a tailored user experience.
- **Scalability:** The use of modern web technologies like TypeScript and React ensures that the application can scale effectively.
- **Integration:** Seamless integration with backend services ensures a smooth user experience.
- **Production-Ready:** Building a production-ready architecture enhances reliability, performance, and maintainability.

Figures 2, 3, and 4 showcase the final user interface developed for the RAG for Resumes platform.

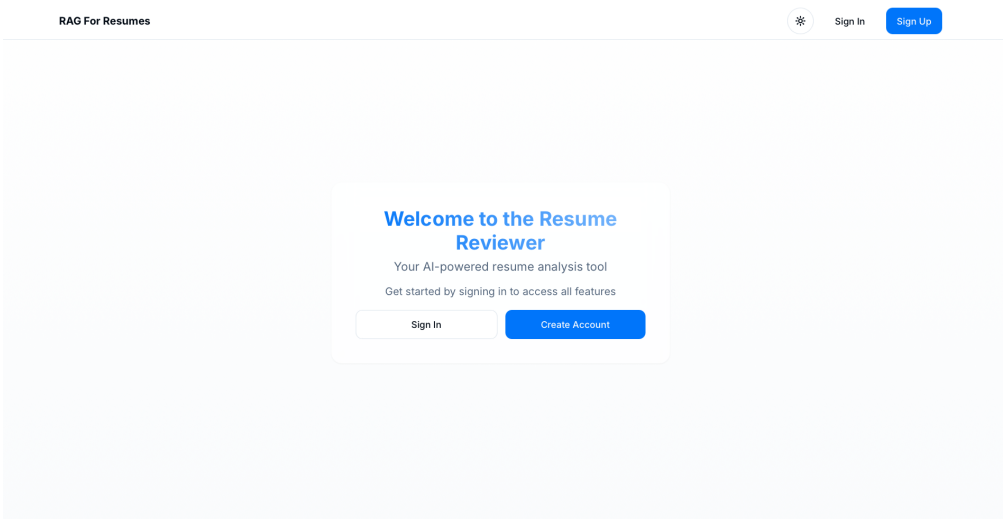


**Figure 2:** Landing page showcasing the sign-in and account creation interface.

# 9 Ethical Considerations

In addition to technical challenges, the project addresses critical ethical dimensions associated with the use of artificial intelligence (AI) in recruitment systems. Given the potential impact on employment opportunities and social equity, particular attention was given to bias mitigation, fairness, transparency, and human oversight.

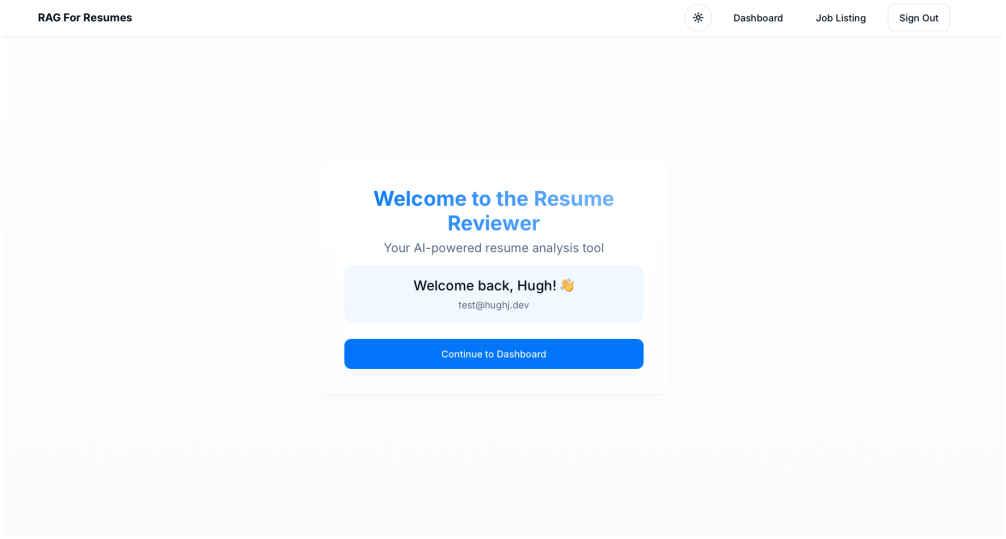


**Figure 3:** Dashboard view after successful login, providing navigation to key features.

- **Bias Mitigation and Model Evaluation:** Recognizing that AI models can inadvertently perpetuate biases embedded in historical data [9], the project implements systematic evaluation of model performance across different demographic groups. Strategies such as balanced retrieval datasets and fairness-aware evaluation metrics are incorporated to minimize unintended biases.
- **Fairness, Transparency, and Explainability:** To foster equitable hiring practices, the system is designed to provide transparent and explainable outputs. The retrieval component of the RAG architecture explicitly surfaces source documents alongside generated responses, thereby supporting interpretable and auditable decision-making processes.
- **Human Oversight and Accountability:** Despite automation efficiencies, the project explicitly maintains a human-in-the-loop framework. Final hiring decisions remain under human authority, with the AI system serving as a decision-support tool rather than an autonomous evaluator, thus ensuring ethical accountability and contextual judgment.

These ethical considerations are integral to the project's design and implementation, ensuring that the RAG system not only enhances recruitment efficiency but also aligns with principles of fairness, transparency, and social responsibility. By addressing these ethical dimensions, the project aims to contribute positively to the evolving landscape of AI in recruitment while

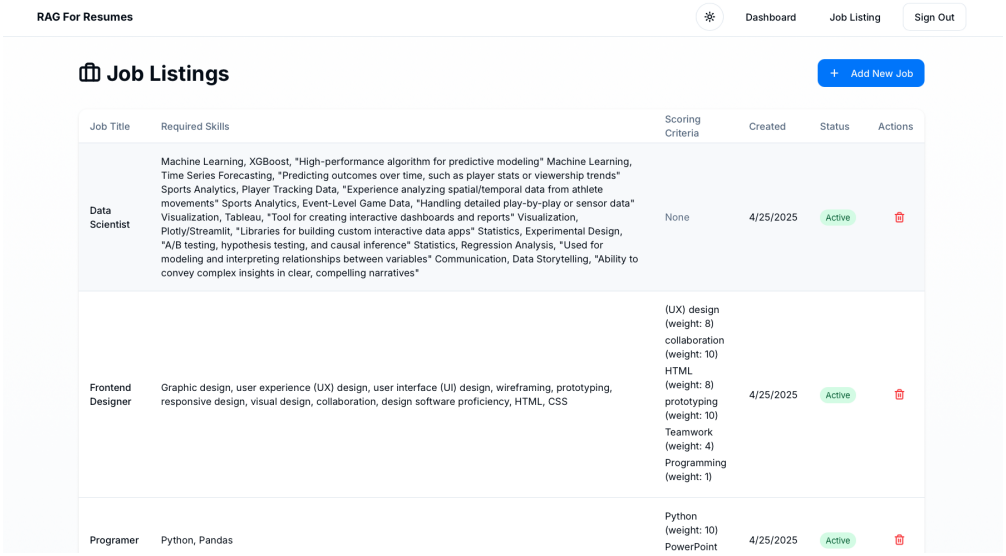


**Figure 4:** Job Listings management interface displaying created jobs, required skills, and scoring criteria.

safeguarding candidate rights and promoting equitable hiring practices.

## 10 Architecture Overview

This system is designed with a modular architecture that ensures security, scalability, and flexibility. As mentioned earlier, users will have to authenticate themselves using Clerk, a third-party authentication service. This integration is crucial for protecting sensitive personal information contained within resumes and maintaining compliance with privacy regulations. After authentication, user can upload multiple or a single resume, and select from a preprogrammed job description. The system will then parse the resumes and job descriptions, extracting relevant information for analysis. The RAG architecture will be employed to retrieve relevant context from the knowledge base (the dataset of resumes) and generate insights based on this context. The retrieved information will be combined with the original input query to provide a comprehensive analysis of how well a candidate's qualifications align with the requirements of a given role. Here is a view of the current architecture:

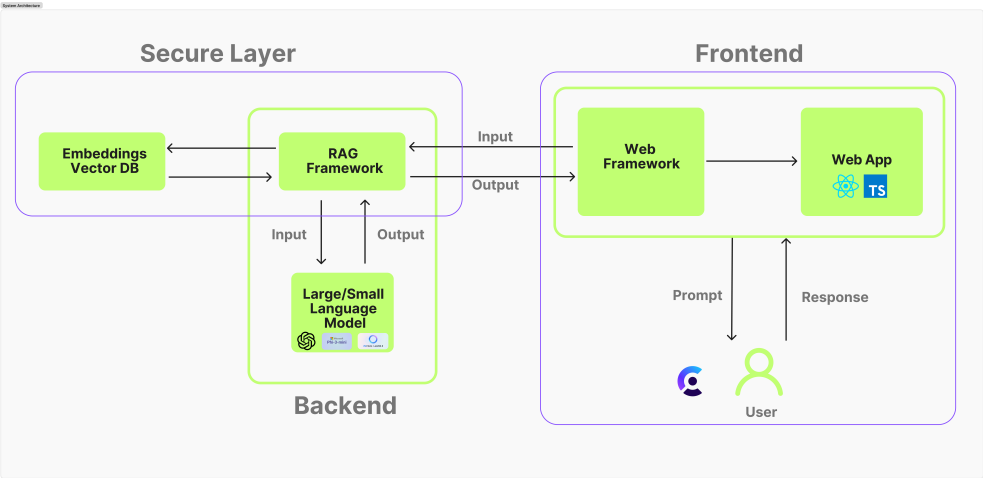


**Figure 5:** Shows the architecture of the system.

# 11 CONCLUSION

This project aimed to create a Retrieval-Augmented Generation (RAG) system for more nuanced, ethical, and privacy-conscious resume analysis and matching. Moving beyond traditional keyword-based methods, the system leverages advanced language models and a large dataset of over 3000 resumes to deliver context-aware candidate-to-job matching. A key innovation is the integration of RAG, which combines data retrieval with generative modeling. This approach enables a deeper understanding of candidate profiles and job descriptions, allowing the system to capture context and semantics rather than relying solely on keyword occurrences. Privacy was a central focus throughout development. Instead of working directly with raw, sensitive personal information, the system uses vector embeddings and a specialized vector database like Pinecone. This method reduces exposure to sensitive data and lowers the risk of breaches, ensuring compliance with privacy regulations and fostering user trust. Beyond privacy, the project also addressed ethical considerations such as bias, fairness, and transparency. It acknowledges the importance of human oversight to maintain integrity and promote equitable hiring practices. The final solution, presented through a Streamlit interface, offers a user-friendly experience for recruiters. Users can input job descriptions or resumes and quickly receive insightful, context-aware matching results. This seamless experience supports the practical integration of the system into real-world recruitment workflows.

# 12 ACKNOWLEDGEMENTS

I would like to thank Dr. Thabet Kacem for his guidance and support throughout this project. His thoughts on the project has been invaluable in shaping the direction and success of this work. I would also like to express my gratitude to my peers and colleagues for their encouragement and feedback, which have greatly contributed to the development of this project. I would also like to thank Dr. Anteneh Girma for his insights on security and privacy, which have been instrumental in ensuring that the system is designed with these critical considerations in mind. Finally, I would like to acknowledge the support of my family and friends, who have been a constant source of motivation and encouragement throughout this journey.

# BIBLIOGRAPHY

[1] S. Rojas-Galeano, J. Posada, and E. Ordoñez, "A bibliometric perspective on ai research for job-résumé matching," *The Scientific World Journal*, vol. 2022, 1–15, 2022. https://doi.org/10.1155/2022/8002363.

[2] A. Gawande. "How semantic search is being used in ai recruitment." CVViZ Blog, April 30, 2024. [Online]. Available: https://cvviz.com/blog/how-semantic-search-used-in-recruitment/.

[3] A. H. Alderham and E. S. Jaha, "Improved candidate-career matching using comparative semantic resume analysis," *Advances in Science, Technology and Engineering Systems Journal*, 2024, Accessed: December 13, 2024. [Online]. Available: https://www.astesj.com/v09/i01/p03/#1639848202927-7c57dd68-5f71.

[4] R. Shan, "A Deep Dive into Vector Stores: Classifying the Backbone of Retrieval-Augmented Generation," *2021 IEEE International Conference on Big Data (Big Data)*, 8831–8833, Dec. 2024. https://doi.org/10.1109/bigdata62323.2024.10825992. [Online]. Available: https://doi.org/10.1109/bigdata62323.2024.10825992.

[5] L. He, P. Tang, Y. Zhang, P. Zhou, and S. Su, "Mitigating privacy risks in retrieval-augmented generation via locally private entity perturbation," *Information Processing & Management*, vol. 62, no. 4, 104150, 2025. https://doi.org/10.1016/j.ipm.2025.104150.

[6] A. Boppana, "Enhancing semantic search in high-dimensional vector spaces: An analysis of sentence embeddings and indexing strategies. "[Online]. Available: https://www.proquest.com/dissertations-theses/enhancing-semantic-search-high-dimensional-vector/docview/3165531406/se-2?accountid=8285.

[7] D. Rothman, *RAG-Driven Generative AI*. Packt Publishing Ltd, Sep. 2024.

[8] L. Silva and L. Barbosa, "Improving dense retrieval models with LLM augmented data for dataset search," *Knowledge-Based Systems*, vol. 294, 111740, Apr. 2024. https://doi.org/10.1016/j.knosys.2024.111740. [Online]. Available: https://doi.org/10.1016/j.knosys.2024.111740.

[9] N. Tilmes, "Disability, fairness, and algorithmic bias in AI recruitment," *Ethics and Information Technology*, vol. 24, no. 2, Apr. 2022. https://doi.org/10.1007/s10676-022-09633-2. [Online]. Available: https://doi.org/10.1007/s10676-022-09633-2.