End-to-End Job Matching Tool: A Multi-Agent Approach to Candidate-Centric Recruitment

1st Syed Ahmed
Dept. of Computer Science
Wilfrid Laurier University
Waterloo, Canada
contact@syedd.com

2nd Emad Mohammed
Dept. of Computer Science
Wilfrid Laurier University
Waterloo, Canada
emohammed@wlu.ca

I. ABSTRACT

The recruitment landscape has shifted toward AI-driven Applicant Tracking Systems (ATS), creating an asymmetry where employers utilize advanced automation while job seekers rely on manual workflows. This paper introduces an end-to-end multi-agent system designed to automate the job-seeking pipeline, from discovery to application submission. Leveraging a supervisor-agent architecture, the system coordinates specialized agents for research, feedback, and job matching using Large Language Models (LLMs) and Retrieval-Augmented Generation (RAG). We evaluate the system on retrieval precision and semantic alignment against human recruiters. Results demonstrate a Mean Precision@10 of 0.879 in job retrieval. However, comparison against human annotations reveals a conservative bias in current LLM matching agents, highlighting the need for fine-tuning to bridge the gap between algorithmic filtering and human intuition.

II. INTRODUCTION

The recruitment landscape has undergone a dramatic transformation in recent years, driven by advances in artificial intelligence and natural language processing. A key driver of this shift has been the widespread adoption of Applicant Tracking Systems (ATS) software platforms designed to automate and streamline hiring workflows. These systems enable organizations to post job openings, automatically screen candidates based on resume content, and generate hiring analytics. Today, approximately 99% of Fortune 500 companies and nearly three-quarters of large enterprises rely on ATS platforms to manage talent acquisition [1]. Increasingly, these systems incorporate AI to evaluate resumes, identify skill matches, and rank candidates, allowing organizations to process high application volumes with greater efficiency and consistency.

While AI-powered ATS platforms and professional networking sites like LinkedIn have enhanced recruitment efficiency for employers, they have simultaneously introduced new challenges for job seekers. Automated keyword-based filtering creates friction for applicants, who must now optimize resumes for algorithmic readability rather than human evaluation. Consequently, candidates face a growing cognitive burden: navigating thousands of job opportunities, manually tailoring resumes and cover letters for each application, and managing uncertain feedback throughout the process. This reveals a fundamental asymmetry while organizations leverage sophisticated AI-driven tools, job seekers continue to depend on largely manual and fragmented workflows.

This paper introduces an end-to-end multi-agent system designed to automate the job-seeking pipeline, from job discovery to application submission. The system employs coordinated AI agents to handle discovery, provide feedback, and score job matches. By automating these processes, the framework aims to reduce cognitive load on candidates while improving the precision of job-applicant matching.

This research contributes to the discourse on AI in recruitment by demonstrating how multi-agent automation can ethically and effectively support job seekers. In doing so, it addresses the imbalance between employer-centric AI innovation and candidate-centered usability, advancing both efficiency and equity in the modern employment ecosystem.

III. RELATED WORK: MULTI-AGENT SYSTEMS IN AI-DRIVEN RECRUITMENT

The integration of Artificial Intelligence and Multi-Agent Systems in the recruitment process has evolved with the advancements of LLMs to address the longstanding challenges in traditional Applicant Tracking Systems (ATS), such as information overload, subjectivity, and lack of transparency [3], [4]. This review takes a look at four key works, looking at both how these systems enhance the job matching process and feedback for seekers as well as reveal gaps in ethical and user-centred design.

A. Early Data Aggregation focused Multi Agent Systems

The foundational challenge in a job recommendation system is unifying data from hundreds of job boards such as LinkedIn, Indeed, Ashby, and Greenhouse. In the JADE-based multi-agent system proposed by Tentua et al. [4] for Indonesian job portals, the system employs two categories of agents: Information gathering agents which scrape, deduplicate, and update a central database with vacancies; and User Agents that query matches based on profiles. A simple two-layer system is proposed with the frontend being a simple UI for notifications and the backend primarily consisting of Agents and the database. The paper reviews recommendation strategies (content-based, collaborative, hybrid) but implements basic rule-based matching without ML.

While this simple system is scalable it has limitations, such as: no quantitative evaluation (precision, recall), vulnerability to scraping blocks, and lack of personalization beyond the profile. As a 2020 baseline study, it highlights the Multi-Agent System's role in data orchestration.

B. NLP and LLM Pipelines

Building on aggregation, the paper "AI for Career Growth" [5] introduces NLP-driven pipelines and semantic understanding of candidate-centric tools. The framework exemplifies this by targeting job seekers with a Streamlit deployed frontend UI. The backend consists of PyPDF2/Selenium which extracts and preprocesses resumes (tokenization, stopwords removal) followed by NLP (NER and Keyword Analysis). Then the information is passed to a Gemini LLM which generates job titles via prompt-based similarity matching, augmented by LinkedIn scrapings for job postings. The paper achieves an approximate 90% F1 accuracy on diverse resumes.

The work surveys 20 prior NLP applications and critiques traditional methods' subjectivity and privacy issues, advocating AI to be deployed locally or on the cloud. However, as this framework acts as a Linear Pipeline rather than a multi-agent system, it lacks modularity. Compared to [4], [5] enhances personalization but inherits scalability bottlenecks as single-threaded processing struggles with high-volume queries in engineering job markets.

C. LLM Based Multi-Agent Frameworks

Recent LLM integrations with Multi-Agent Systems help overcome single LLM limitations such as opacity and non-adaptability, as well as include explainable, multi-perspective decisions. In [2], a user-centred multi-agent system study features three agents: Recruiter Agent (objective scoring of resume-job fit), Mentor (guidance on improvements/interview prep) and Moderator (synthesis/resolution using LLM as a judge). The chatbot outperforms traditional ATS systems in actionability (+34), trust (+30) and fairness (+40).

Complimenting this, the 2025 RAG-LLM Multi-Agent System proposed in [3] uses four modular Agents in CrewAI/LangChain: Extract (Structured parsing agent), Evaluator (role-specific scoring via ChromaDB RAG for external data like certifications/company policies), Summarizer (sub-agents simulating CEO/CTO/HR for collaborative insights), and Formatter (structured outputs). This approach surpasses single LLM baselines on metrics such as Pearson correlation (0.84 vs 0.70) on 105 anonymized HR resumes.

In conclusion, these frameworks advance beyond early aggregation methods like Tentua et al.'s JADE-based system and the linear NLP pipelines described in "AI for Career Growth" by utilizing true agentic collaboration. Recent works, such as Bhattacharya and Verbert's user-centered chatbot and Lo et al.'s RAG-LLM screening tool, chart the evolution of Multi-Agent Systems in recruitment from reactive foundations to transparent, LLM-driven architectures. However, these existing solutions primarily focus on employer-side screening or isolated user tasks. This study differentiates itself by proposing a comprehensive end-to-end framework specifically for the job seeker, integrating discovery, feedback, and application into a unified loop that prioritizes candidate utility over recruiter efficiency.

IV. Methodology

The end-to-end job agent framework streamlines the application process through three interconnected layers. The frontend layer provides the user interface where candidates interact with AI agents and access key information, including saved job listings, compatibility scores, and resume documents. The backend layer manages authentication, data persistence through PostgreSQL databases, and file storage via Amazon S3 buckets, all hosted on Supabase infrastructure. The agent layer contains the core AI components responsible for task execution and decision-making. Figure 1 presents a simplified representation of the complete system architecture.

A. Agent Architecture

Multi-agent systems have emerged as a significant area of research in both academic and commercial domains. Studies comparing single large language models (LLMs) against multi-agent architectures in simulated human reasoning tasks demonstrate substantial performance advantages, with multi-agent systems achieving accuracy rates of 87.5% compared to 50% for single

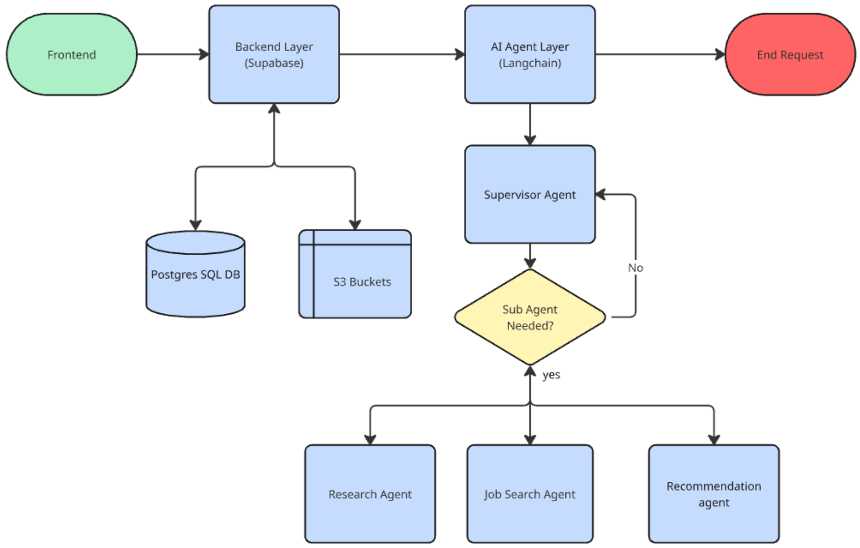

Fig. 1. A simplified outline of the System Architecture

LLMs [6]. The proposed system implements a supervisor pattern, wherein a central coordinating agent manages task delegation to specialized sub-agents based on user queries.

The framework comprises four distinct agents, each with dedicated responsibilities and tool access as detailed in Table I. All agents follow the Reasoning and Acting (ReAct) pattern and are implemented using open-source frameworks, including LangChain and LangGraph. LangSmith was employed for testing and evaluation purposes, and was later containerized via Docker and deployed as a self-hosted minimalistic server.

TABLE I
AGENT OVERVIEW

Agent Name	**Agent Description**	**External Tools**
Supervisor Agent	Delegates tasks and coordinates sub-agents	Feedback Agent, Job Search Agent, Research Agent
Research Agent	Conducts company research and information retrieval	Exa Web Search API
Feedback Agent	Generates personalized user feedback and recommendations	N/A
Job Matcher Agent	Identifies and ranks relevant job opportunities	Google Job Search API, Vector DB Retriever

B. Agent Tools

Tools extend agent capabilities by providing access to external data sources and enabling interaction with real-world systems. The framework integrates three primary external tools: Exa Web Search, Google Search API, and Chroma Vector Database Retriever for Agentic Retrieval-Augmented Generation (RAG).

Exa Web Search: Exa provides a specialized API optimized for AI agent search queries. At the time of implementation, Exa demonstrated superior benchmarking performance for AI-driven search tasks while offering a generous free-tier access model. These characteristics made it the preferred choice for the Research Agent, enabling real-time retrieval of company-specific information while reserving standard Google Search for broader query tasks.

Google Search API: Google Search leverages Boolean operators (e.g., AND, OR) to construct precise search queries with keyword constraints. This functionality proves particularly valuable for job discovery based on user-specified criteria. For instance, the query `site:ashbyhq.com ("AI Engineer" OR "FullStack" OR "Software") AND "san` francisco" retrieves all job postings from AshbyHQ, a widely adopted ATS platform, matching the specified roles and location. The Google Search tool accepts Boolean query strings and executes filtered searches across multiple job boards, including greenhouse.io, ashbyhq.com, jobs.lever.co, and careers.workable.com. The tool returns structured results containing job titles, company names, locations, compatibility scores, and application URLs.

Chroma Vector Store and Retriever: Retrieval Augmented Generation is a commonly employed technique in order to increase the performance of an LLM by giving it additional relevant context typically stored in a Vector or Graph Database. This paper utilizes a vector database, self-hosted on ChromaDB, an open-source vector store. Embeddings of datasets of publicly available job posting datasets sourced from sites such Kaggle. An open source embedding model "all-MiniLM-L6-v2" was used to generate embeddings in a 384-dimensional dense vector space.

C. Frontend Architecture and User Interface

The frontend application provides an intuitive user interface for candidates to interact with the multi-agent system through a streamlined workflow from registration to job application.

User flow and interface components: Figure 2 illustrates the complete user journey through the application. The workflow consists of the following stages:

Authentication and Onboarding: Upon initial access, users encounter an authentication checkpoint. New users are directed through a registration and onboarding process that captures essential information including professional background, resume upload, and relevant skills. This onboarding data forms the basis of personalized job matching and resume optimization.

Primary Interface Screens: The application comprises four primary interface components; screenshots of the UI can be seen in Figure 3:

1) **Homepage**: Serves as the central hub, providing a quick access screen to enter the chat as well as other pages via the sidebar.
2) **Resumes**: A dedicated interface for resume management, allowing users to upload, view, edit, and manage multiple resume versions. This component displays stored resumes retrieved from the S3 bucket storage system.
3) **Chat Interface**: The primary interaction point with the AI agent system. Users engage in natural language conversations to specify job search criteria, request company research, receive personalized feedback, and refine search parameters.
4) **Job Postings**: A table-based view displaying saved job opportunities identified by the system.

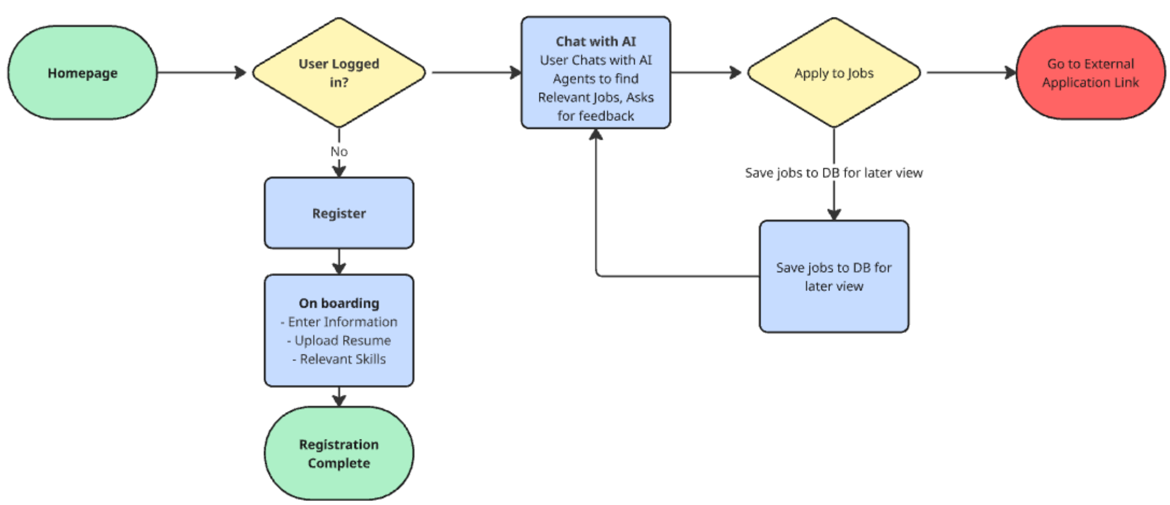

Fig. 2. Typical user flow upon initial login into system

D. Data Management

The system implements a minimal data storage architecture that balances functionality with privacy considerations. The data layer employs a PostgreSQL relational database hosted on Supabase, an open-source Backend-as-a-Service (BaaS) platform, complemented by Amazon S3-compatible object storage for document management.

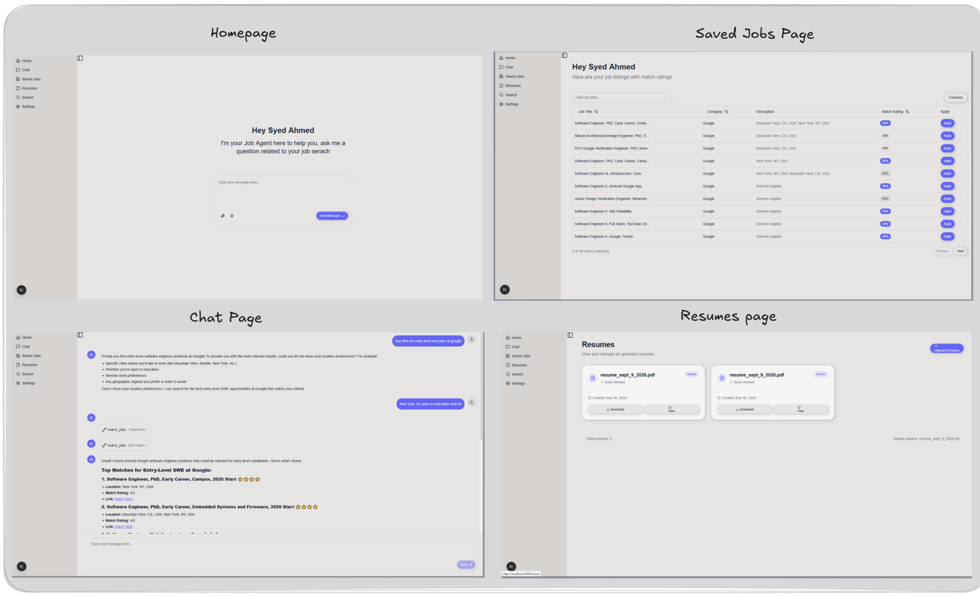

Fig. 3. Different pages within the user interface

E. Database Architecture

The PostgreSQL database maintains four primary tables as illustrated in Figure 4. The `users` table stores essential profile information including name, email, professional background, and skill sets collected during onboarding. The `resumes` table contains metadata linking user identities to their uploaded documents, with actual resume files stored separately in S3 buckets to optimize database performance. The `jobs` table persists discovered job opportunities, storing structured fields such as job title, company name, location, compatibility scores, posting dates, and application URLs. The `chat_history` table maintains conversation threads between users and the AI agents, enabling context-aware interactions across sessions while supporting system evaluation and continuous improvement.

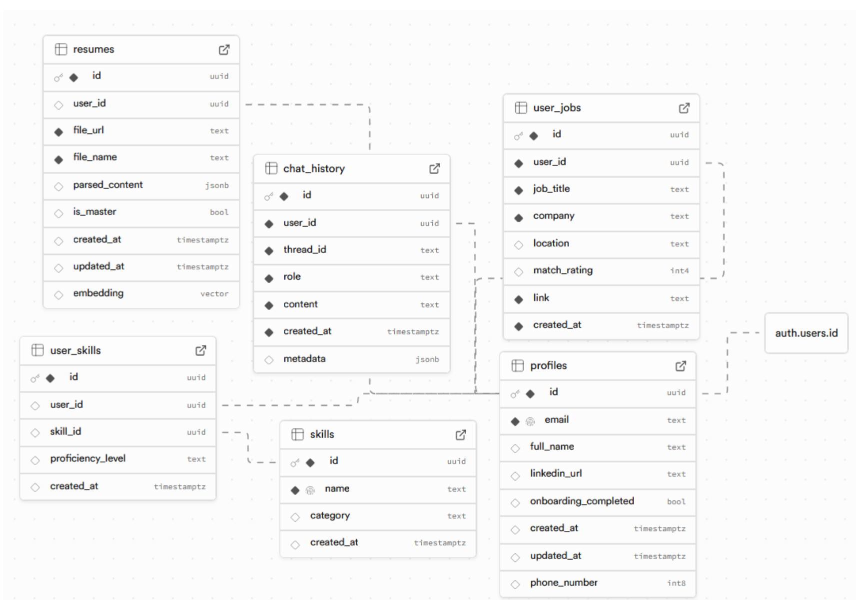

Fig. 4. Database Schema

V. Evaluation and Results

To assess the effectiveness of the end-to-end framework, we conducted a two-fold evaluation focusing on: (1) The retrieval precision of the Research Agent and Vector Store, and (2) The semantic alignment of the Job Matcher Agent against human recruiters.

A. Retrieval Quality

We evaluated the retrieval module using a publicly available job dataset on Hugging Face [8]. We simulated 43 synthetic user personas across varying domains (Technology, Business, and Location-Specific) and performed top-k retrieval against a vector store of job postings embedded with `all-MiniLM-L6-v2`, an open-source embedding model.

Precision and Ranking: The system demonstrated exceptional performance in surfacing relevant documents. We achieved a **Mean Precision@10 of 0.879**, indicating that approximately 9 out of 10 recommended jobs were relevant to the user query. Furthermore, the **Mean Reciprocal Rank (MRR) of 0.934** confirms that the most relevant document appeared at the first rank in nearly all test cases.

Recall vs. Corpus Size: The system exhibited a low Recall@10 of 0.0019. This is an artifact of the corpus density; for generic queries such as "Account Executive," the dataset contained over 21,000 relevant documents. As the system is designed for user utility rather than exhaustive search, we prioritized Precision over Recall to minimize candidate cognitive load.

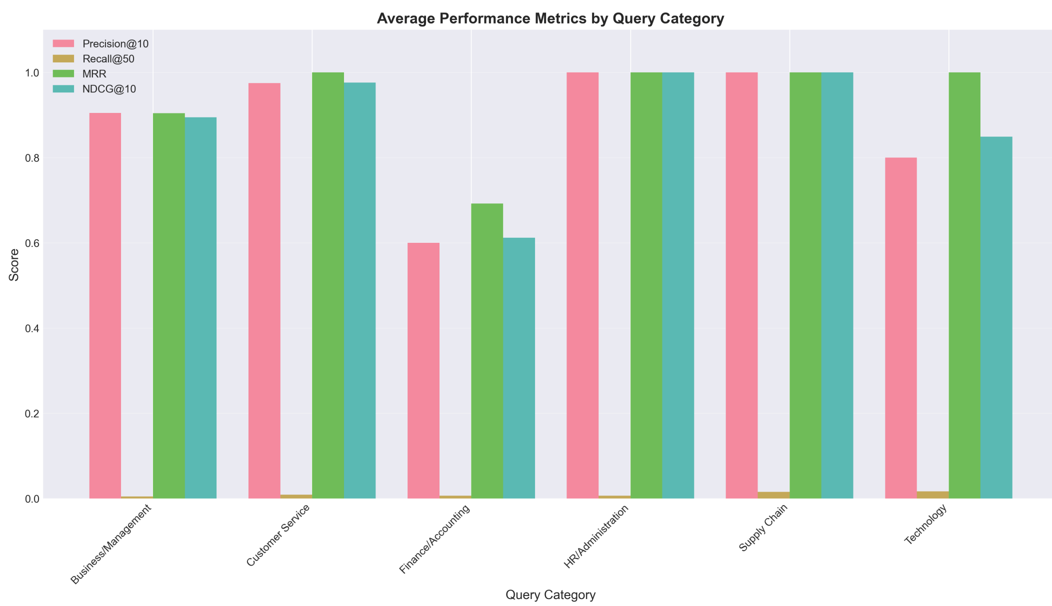

Fig. 5. Retrieval performance across different query domains. Note the high performance in Business/Tech versus the drop in Location-specific queries.

Domain Sensitivity: A breakdown of query performance (see Fig. 5) reveals that the vector retrieval is highly effective for Role and Skill-based queries but struggles with geolocation.

- **Technology Queries:** High robustness (Avg Precision: 0.80).
- **Business Queries:** Excellent performance (Avg Precision: 0.88).
- **Location Queries:** Performance degradation (Avg Precision: 0.61).

Notably, Query 36 ("Accountant role in Petaling Jaya") yielded 0.0 precision, highlighting a limitation in the embedding model's ability to attend to specific geographic constraints compared to traditional boolean filtering.

B. Matching Accuracy and Alignment

To evaluate the *Job Matcher Agent*, we utilized the *Vacancy-Resume Matching Data* [7], comparing the Agent's compatibility scores (1-5 Likert scale) against ground-truth annotations provided by human HR specialists across 145 resume-vacancy pairs. A similar user ranking agent was prepared for the evaluation as the original in the application had access to external tools such as web search.

Conservative Bias: The evaluation revealed a divergence in scoring philosophy. The Agent exhibited a distinct "conservative bias," consistently scoring candidates approximately 0.77 points lower than human annotators (MAE = 1.317). While human scores followed a normal distribution, the Agent's scores clustered heavily around "Low-Medium Fit" (Score 2).

Correlation: Consequently, the Pearson correlation coefficient ($r = -0.032$) indicates a lack of linear alignment. However, this does not necessarily imply failure; rather, it suggests the zero-shot Agent operates as a strict "Hard Filter," rejecting candidates who do not possess a near-perfect keyword match, whereas human recruiters apply intuition to identify potential.

C. Discussion

The results validate the hybrid architecture. The *Retrieval Agent* effectively reduces the search space with high precision ($> 87\%$), ensuring users are not flooded with irrelevant noise. Meanwhile, the *Job Matcher Agent* currently functions as a strict gatekeeper. Future work will focus on Fine-Tuning the matching agent to better emulate the "leniency" and intuition of human recruiters regarding transferable skills.

VI. Conclusion

This paper presented an end-to-end multi-agent framework designed to rebalance the recruitment ecosystem by empowering job seekers with enterprise-grade automation. By coordinating specialized agents for research, feedback, and job matching, the system successfully alleviates the cognitive burden of the manual application process.

Our evaluation demonstrates that while the retrieval architecture achieves high precision ($> 87\%$) in identifying relevant opportunities, the system is not without limitations. The Zero-Shot LLM-based matching agent exhibits a distinct conservative bias compared to human recruiters, operating as a strict filter rather than applying intuitive judgment on transferable skills. Additionally, the retrieval module showed degraded performance when handling specific geolocation constraints compared to broad role-based queries.

Future work will address these gaps by implementing a "Human-in-the-Loop" feedback mechanism to fine-tune agent scoring models and improving the geographic sensitivity of the embedding strategy. To foster collaboration and accelerate development in candidate-centric AI, the full source code and datasets have been open-sourced for further development [9]. We invite the community to contribute to the refinement of these tools to create a more equitable labor market.

References

[1] Select Software Reviews, "Applicant Tracking System Statistics," *SelectSoftwareReviews.com*. Available: https://www.selectsoftwarereviews.com/blog/applicant-tracking-system-statistics. Accessed: Oct. 22, 2025.

[2] A. Bhattacharya and K. Verbert, "Let's get you hired: A job seeker's perspective on multi-agent recruitment systems for explaining hiring decisions," *arXiv preprint arXiv:2505.20312*, 2025.

[3] F. P.-W. Lo, J. Qiu, Z. Wang, H. Yu, Y. Chen, G. Zhang, and B. Lo, "AI hiring with LLMs: A context-aware and explainable multi-agent framework for resume screening," *arXiv preprint arXiv:2505.20312*, 2025.

[4] M. N. Tentua, A. S. R. Siagian, and A. P. R. Pinem, "The multi-agent system for job recommendation," in *IOP Conference Series: Materials Science and Engineering*, vol. 835, no. 1, p. 012039, 2020. Available: https://iopscience.iop.org/article/10.1088/1757-899X/835/1/012039/pdf

[5] K. S. Kumar, P. Srihari, and C. J. Raman, "AI for career growth: Advanced resume analysis and LinkedIn scraping for personalized job recommendations," 2024. Available: https://www.researchgate.net/publication/387927216

[6] K. Sreedhar and L. Chilton, "Simulating human strategic behavior: Comparing single and multi-agent LLMs," *arXiv preprint arXiv:2402.08189*, 2024.

[7] N. Vanetik and M. Litvak, "Vacancy-resume matching dataset," *GitHub repository*, 2018. [Online]. Available: https://github.com/NataliaVanetik/vacancy-resume-matching-dataset

[8] Azrai99, "Job-dataset," *Hugging Face*, 2023. [Online]. Available: https://huggingface.co/datasets/azrai99/job-dataset

[9] S. Ahmed, "End-to-End Job Search Agent," *GitHub repository*, 2024. [Online]. Available: https://github.com/Syed-Ahmed02/E2E-Job-Search-Agent