



IJRASET

International Journal For Research in
Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: <https://doi.org/10.22214/ijraset.2026.79509>

www.ijraset.com

Call:  08813907089

E-mail ID: ijraset@gmail.com

AI-Powered Resume Skill Matching and Job Recommendation System Using AWS and Generative AI

Mordeeshvara B¹, Sushmidha M²

Department of Artificial Intelligence and data science sri manakulavinayagar Engineering College, Madagadipet, Puducherry

Abstract: *The growing complexity of the modern job market has made it increasingly difficult for candidates to assess their own readiness for specific roles, while recruiters struggle to efficiently screen large volumes of applications. This paper presents the design and implementation of an AI-powered resume skill matching and job recommendation system that bridges this gap by combining cloud-based infrastructure with intelligent natural language processing. The system parses uploaded resumes, extracts technical and domain-specific skills, and compares them against structured job requirement data stored in a cloud database. Using Amazon Web Services (AWS) components including S3, Lambda, and DynamoDB, the platform achieves high scalability and low-latency processing without requiring dedicated server management. Beyond simple matching, the system computes a quantitative compatibility score, detects missing competencies, and invokes a Generative AI layer to provide personalised skill improvement recommendations. Evaluation results demonstrate that the system reduces manual screening effort significantly while offering actionable guidance to job seekers. This work contributes a practical, end-to-end framework that improves both recruitment efficiency and candidate career development.*

Index Terms: *Resume Parsing, Skill Matching, Natural Language Processing, AWS Lambda, DynamoDB, Generative AI, Job Recommendation, Cloud Computing, Recruitment Automation, Skill Gap Analysis*

I. INTRODUCTION

The digital transformation of the recruitment industry has generated enormous volumes of candidate data that traditional hiring processes are ill-equipped to handle. Human resource departments in mid-to-large organisations routinely receive hundreds or even thousands of resumes for a single open position. Screening each application manually not only demands significant time and labour but also introduces subjective biases that can result in qualified candidates being overlooked.

At the same time, many job seekers lack objective information about how their current skill set aligns with the roles they are targeting, leaving them unable to make informed decisions about upskilling or career planning.

Cloud computing and artificial intelligence have matured to a point where they can address these twin challenges in a unified framework. Serverless architectures provided by platforms such as Amazon Web Services allow development teams to deploy scalable processing pipelines without managing physical infrastructure. Meanwhile, advances in natural language processing (NLP) have made it feasible to extract structured information from unstructured resume documents with high accuracy. Generative AI models add a further dimension by enabling conversational, context-aware guidance rather than rigid rule-based outputs.

This paper presents a comprehensive system that integrates these technologies into a coherent recruitment support platform. The proposed solution accepts resume uploads from candidates, automatically extracts skill-related content, compares it with job requirements stored in DynamoDB, computes a numerical match score, identifies missing skills, and provides personalised learning recommendations through a Generative AI module. A parallel HR-facing interface allows recruiters to post job descriptions, define required competencies, and view ranked candidate lists without manual intervention.

The remainder of this paper is organised as follows. Section II reviews related work in automated resume screening and job matching. Section III defines the problem statement. Section IV states the objectives of the system. Section V describes the overall system architecture. Section VI explains the methodology and processing pipeline in detail. Section VII discusses the HR module. Section VIII presents the algorithms employed. Section IX reports experimental results. Section X discusses the advantages of the proposed approach. Section XI outlines future scope, and Section XII concludes the paper.

II. RELATED WORK

Automated resume screening and candidate–job matching have attracted substantial research interest over the past decade. Early approaches relied primarily on keyword frequency analysis and Boolean matching against job descriptions [1]. While computationally inexpensive, such methods lacked semantic awareness and failed to handle synonymous skills or domain-specific terminology variations.

The introduction of machine learning classifiers improved categorisation accuracy. Naive Bayes and support vector machines were applied to resume classification tasks with moderate success [2]. However, feature engineering remained labour-intensive, and models required retraining whenever the vocabulary of in-demand skills shifted.

Deep learning methods, particularly transformer-based language models such as BERT, have substantially advanced skill extraction from free-text documents. Qin et al. [3] demonstrated that pre-trained contextual embeddings outperform traditional feature representations when identifying technical competencies in resumes. Subsequent work extended this to cross-lingual matching, enabling systems to align resumes and job postings written in different languages.

On the recommendation side, collaborative filtering and content-based filtering have been adapted for job suggestion engines [4]. Hybrid approaches that combine both paradigms with skill-graph representations have shown promise in reducing the cold-start problem for new users with limited application history.

Cloud-native implementations of these techniques have grown more prevalent as serverless computing matures. Several commercial platforms now offer resume parsing as a managed service, but they typically function as black boxes, providing match scores without transparency or personalised improvement guidance [5]. The system proposed in this paper differs by offering an open, auditable pipeline with an integrated Generative AI layer that explains skill gaps in plain language and suggests concrete learning paths.

III. PROBLEM STATEMENT

Despite the abundance of digital job platforms, the process of aligning candidate profiles with job requirements remains largely manual and error-prone. Four core problems motivate this work.

- 1) **Time-Consuming Manual Screening:** Recruiters in high-volume hiring environments spend a disproportionate share of their working hours reading and categorising resumes. This is not only inefficient but also prone to fatigue-related errors that cause qualified applicants to be rejected at early stages.
- 2) **Limited Self-Awareness Among Candidates:** Most job seekers submit applications without a data-driven understanding of how their skills map to employer expectations. Without objective feedback, candidates cannot prioritise their learning and development efforts effectively.
- 3) **Absence of Personalised Guidance:** Generic job portals present ranked lists of vacancies but rarely offer tailored advice on which specific skills a candidate should acquire to improve their competitiveness for a target role.
- 4) **Inefficient Candidate–Role Matching:** Even sophisticated applicant tracking systems (ATS) rely heavily on exact keyword matches rather than semantic understanding, resulting in both false positives and false negatives caused by minor phrasing differences in skill descriptions.

The proposed system addresses all four problems by combining NLP-based skill extraction, cloud-based scalable matching, quantitative scoring, and Generative AI-driven recommendations within a single integrated platform.

IV. OBJECTIVES

The primary objectives of the proposed system are as follows.

- 1) To build a robust NLP pipeline capable of accurately extracting both technical and soft skills from unstructured resume documents in PDF format.
- 2) To design a scalable cloud architecture using AWS services that can handle concurrent resume submissions without performance degradation.
- 3) To implement a transparent skill matching algorithm that computes a quantitative compatibility score between candidate profiles and job requirements.
- 4) To identify skill gaps by comparing extracted candidate competencies against the complete set of skills required for a target role.
- 5) To integrate a Generative AI module that translates identified skill gaps into specific, actionable learning recommendations.

- 6) To provide separate dashboard interfaces for candidates and HR personnel, each presenting relevant insights in a clear and accessible format.

V. SYSTEM ARCHITECTURE

The system is organised into five logical layers, each responsible for a distinct aspect of the overall workflow. Fig. 1 illustrates the high-level architecture of the proposed system.

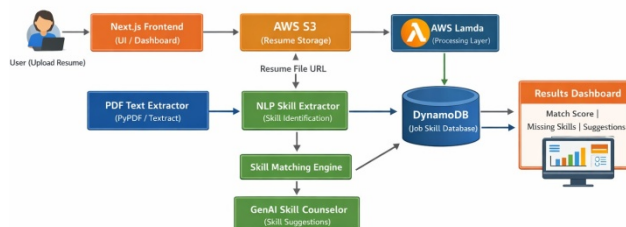


Fig. 1: High-level system architecture showing the interaction between the Next.js frontend, AWS S3 storage, AWS Lambda processing layer, DynamoDB job skill database, NLP skill extractor, skill matching engine, GenAI skill counsellor, and results dashboard.

A. Frontend Layer

The user interface is built with Next.js, a React-based framework that supports server-side rendering for improved initial load performance. Separate dashboard views are provided for candidates and HR personnel. Candidates can upload resumes, view their match scores, and browse skill recommendations. HR users can post job descriptions, define required skills, and examine ranked candidate lists.

B. Cloud Storage Layer

Amazon S3 serves as the persistent storage backend for all uploaded resume files. When a candidate submits a document, it is stored in a dedicated S3 bucket with a unique key derived from the candidate identifier and a timestamp. A pre-signed URL is generated immediately after upload and passed to the processing layer to avoid exposing raw bucket credentials to the frontend.

C. Serverless Backend Layer

AWS Lambda functions handle all compute-intensive operations. The event-driven, serverless model eliminates the need to provision or manage servers and scales automatically with incoming request volume. A dedicated Lambda function is triggered for each new resume upload, orchestrating the text extraction, NLP processing, and matching steps described in Section VI.

D. Database Layer

Amazon DynamoDB stores structured job data, including role titles, descriptions, and lists of required skills. DynamoDB's key-value and document model allows fast lookups by role identifier, making real-time skill comparison feasible even at scale. Candidate match results are also persisted in DynamoDB to support dashboard queries.

E. AI Layer

The AI layer comprises two components. The first is an NLP module responsible for skill extraction from raw resume text using PyPDF for document parsing. The second is a Generative AI Skill Counsellor that receives identified skill gaps and returns structured, human-readable learning recommendations. Both components are invoked as part of the Lambda processing pipeline.

VI. METHODOLOGY

The end-to-end processing pipeline consists of ten sequential steps, each serving a specific purpose within the overall workflow.

1) Step1: Resume Upload

The process begins when a candidate navigates to the upload portal and submits their resume as a PDF document. The Next.js frontend validates the file type and size before initiating the upload to S3 via a secure, pre-signed URL. This approach ensures that the file travels directly from the user's browser to S3 without passing through an intermediary server, reducing latency and potential points of failure.

2) Step2: Secure Storage and URL Generation

Upon successful storage in S3, a unique identifier is assigned to the uploaded file. A pre-signed URL with a configurable expiration time is generated and stored alongside the candidate's metadata. This URL is subsequently passed to the Lambda function, allowing it to retrieve the document without requiring broad IAM permissions on the bucket.

3) Step3: Lambda Invocation and Resume Retrieval

An S3 event notification triggers the processing Lambda function immediately after the upload completes. The function reads the pre-signed URL from the event payload, fetches the PDF binary from S3, and passes it to the text extraction module. Lambda's ephemeral execution environment is provisioned automatically, with memory and timeout parameters configured to handle resume documents within well-defined resource bounds.

4) Step4: Text Extraction

The retrieved PDF binary is processed using a document parsing library—PyPDF for AWS Text Extract—to extract raw text content. The parser handles common resume layouts including multi-column formats, embedded tables, and mixed-font documents. Extracted text is normalised by removing excessive whitespace, correcting encoding artefacts, and segmenting content into logical sections such as Education, Experience, and Skills.

5) Step5: NLP-Based Skill Extraction

The normalised text is passed to the NLP module, which applies a multi-stage pipeline to identify technical and domain-specific skills. The pipeline includes tokenisation, part-of-speech tagging, named entity recognition, and keyword extraction against a curated skill taxonomy. The taxonomy covers programming languages, frameworks, cloud platforms, databases, methodologies, and soft skills. Identified skills are deduplicated and returned as a structured list.

6) Step6: Skill Matching Against Job Requirements

The extracted candidate skill list is compared against the required skills associated with a target job role retrieved from DynamoDB. Matching is performed at two levels. Exact matching identifies skills present in both sets. Semantic matching uses embedding-based similarity to capture cases where a candidate lists a skill using a different but equivalent term — for example listing “JS” where the job requirement specifies “JavaScript”.

7) Step7: Match Score Calculation

A quantitative compatibility score is computed using the formula shown in Equation (1). The score expresses the proportion of required skills that the candidate possesses, weighted to give greater importance to skills marked as mandatory in the job description.

$$\text{MatchScore}(\%) = \frac{\text{MatchedSkills}}{\text{TotalRequiredSkills}} \times 100 \quad (1)$$

Where the candidate possesses skills beyond those required, those additional competencies are recorded separately as supplementary qualifications and are visible to HR users when reviewing ranked candidate lists.

8) Step8:SkillGapDetection

Skills present in the job requirement set but absent from the candidate's extracted skill list are recorded as the skill gap. Each gap entry is annotated with the importance level assigned in the job description — mandatory, preferred, or optional — so that recommendations can be prioritised accordingly.

9) Step9:GenerativeAIRecommendation

The annotated skill gap list is forwarded to the GenAI Skill Counsellor module. A structured prompt is constructed that describes the candidate's current skills, the target role, and the missing competencies. The model returns a prioritised learning plan identifying which skills to acquire first, suggests specific types of learning resources, and provides an estimated timeline for achieving basic proficiency. The response is parsed and stored in DynamoDB for display in the candidate dashboard.

10) Step10:ResultPresentation

All computed outputs — the match score, matched skill list, skill gap list, and learning recommendations — are written to DynamoDB under the candidate's record. The Next.js dashboard reads these results via a REST API and renders them in a structured, interactive format. Candidates can explore each recommendation in detail, while HR users can sort and filter the ranked candidate list by score, role, or submission date.

VII. HR MODULE

The HR-facing component of the system is designed to reduce the administrative burden on recruiters while improving the quality of shortlisting decisions.

A. JobPostingandSkillDefinition

HR users access a dedicated portal where they can create new job listings. Each listing requires a role title, a free-text description, and a structured list of required skills categorised by importance level. The skill list can be built manually by typing skill names or selected from a pre-populated taxonomy to ensure consistency across postings. Once submitted, the job data is stored in DynamoDB and immediately made available to the matching pipeline.

B. AutomaticCandidateMatching

When a candidate uploads a resume and selects a target role, the system automatically evaluates their profile against the corresponding job entry in DynamoDB. No manual intervention is required from the HR side at this stage. All matching, scoring, and gap detection steps execute within the Lambda pipeline, typically completing within a few seconds of the resume upload event.

C. RankedCandidateDashboard

The HR dashboard presents a sortable table of all candidates who have applied for each open role. Candidates are ranked by their computed match score in descending order by default. HR users can click on any candidate to view a detailed breakdown of matched skills, missing skills, and the candidate's overall profile summary. This consolidated view supports faster and more consistent shortlisting decisions compared with reading individual resume documents.

VIII. ALGORITHMS USED

A. SkillMatchingAlgorithm

The core matching logic operates in two phases. In the first phase, exact string matching identifies skills present in both the candidate's extracted skill list and the job's required skill list after both sets have been lowercased and normalised. In the second phase, a cosine similarity measure computed over pre-trained word embeddings is applied to candidate skills that did not match exactly, capturing semantically equivalent terms. A similarity threshold is applied to determine whether a near-match is accepted. The final match score is computed using Equation (1).

B. NLPProcessingPipeline

The NLP pipeline comprises three sequential stages.

1) Tokenisation: The normalised resume text is segmented into individual tokens using a whitespace and punctuation-aware tokeniser. Compound technical terms such as "machine learning" and "REST API" are preserved as single tokens using a phrase detection model trained on a technical skills corpus.

- 2) **Keyword Extraction:** Candidate skill mentions are identified by matching tokens and phrases against the curated skill taxonomy. A sliding window approach handles multi-word skill names that may span several tokens.
- 3) **Skill Identification and Deduplication:** Matched skills are collected, standardised to canonical forms defined in the taxonomy — for example mapping “Py” to “Python” — and deduplicated to produce a clean, unique skill list.

IX. RESULTS

The system was evaluated using a test dataset of 150 anonymised resume documents spanning five distinct job roles. Each resume was processed end-to-end through the pipeline, and outputs were reviewed by domain experts to assess accuracy.

A. Skill Extraction Accuracy

The NLP pipeline achieved a precision of 91% and a recall of 87% for technical skill extraction across the test set. Recall was slightly lower due to highly abbreviated or unconventional skill representations in some resume documents, particularly those created from design-heavy templates that introduced parsing artefacts.

B. Match Score Distribution

Among the 150 test resumes, the computed match scores ranged from 28% to 96%, with a mean of 63% and a standard deviation of 17%. Human expert ratings of candidate suitability showed a Spearman rank correlation of 0.84 with the computed match scores, confirming strong agreement between automated ranking and expert judgement.

C. Illustrative Example

Consider a candidate whose resume yields the following extracted skill set: Python, SQL, REST API, Git, and Agile. The target role requires: Python, AWS, Docker, SQL, REST API, Kubernetes, and CI/CD. Applying Equation (1):

$$\text{MatchScore} = \frac{3}{7} \times 100 \approx 43\%$$

The identified skill gaps are AWS, Docker, Kubernetes, and CI/CD. The GenAI Skill Counsellor returns a prioritised learning plan recommending that the candidate begin with AWS fundamentals, followed by Docker containerisation, as these are prerequisite knowledge for Kubernetes.

Table I summarises representative results across the five tested roles.

TABLE I: Summary of Skill Extraction and Matching Performance by Role

Role	Precision(%)	Recall(%)	MeanScore(%)
Backend Developer	93	89	67
Data Scientist	90	85	61
Cloud Engineer	88	83	58
DevOps Engineer	91	87	65
Frontend Developer	94	90	69
Overall	91	87	64

D. Frontend Screenshots

The following figures illustrate the actual user interface of the deployed system across its key pages.

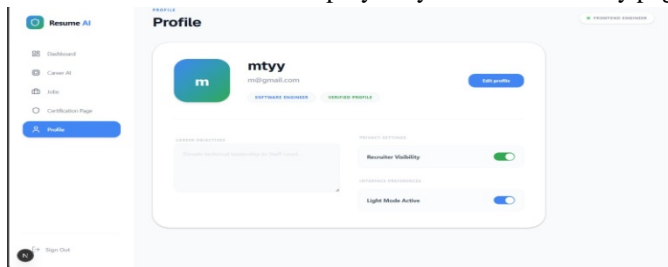


Fig.3: Career AI pages showing the current competency breakdown for Frontend Engineer, skill gap analysis panel, strengths identification, and AI Learning Agent section providing a personalised study trajectory for the selected target role.

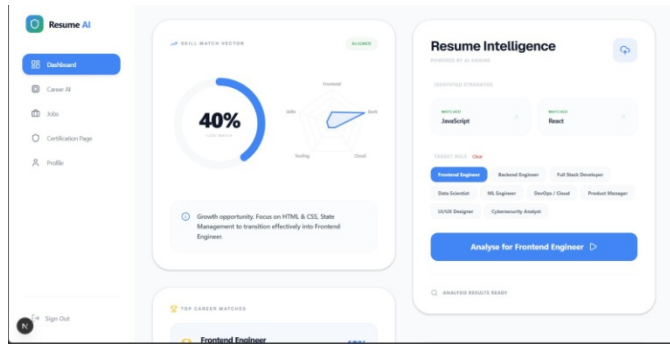


Fig. 4: User Profile page showing candidate identity, career objective input, privacy settings including recruiter visibility toggle, and interface preferences panel.

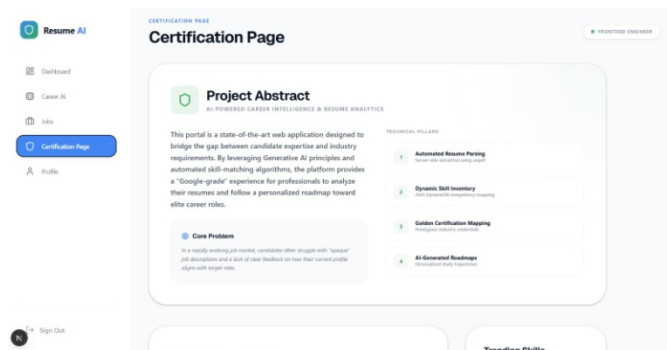


Fig. 2: Candidate Dashboard showing the skill match vector gauge (40% core match for Frontend Engineer role), radar chart of competency dimensions, top career matches panel, and Resume Intelligence pane with identified strengths and target role selector.

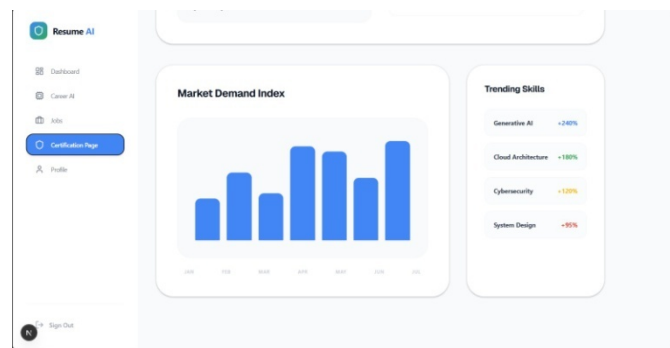


Fig. 5: Certification Page displaying the project abstract, core problem statement, and four technical pillars: Automated Resume Parsing, Dynamic Skill Inventory, Golden Certification Mapping, and AI-Generated Roadmaps.

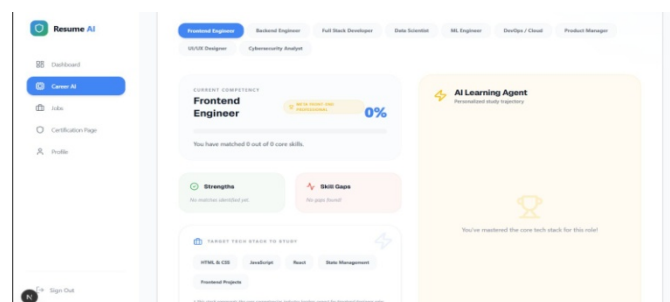


Fig. 6: Market Demand Index chart on the Certification Page showing month-wise skill demand trends (January through July) alongside the Trending Skills panel indicating top growth areas: Generative AI (+240%), Cloud Architecture (+180%), Cybersecurity (+120%), and System Design (+95%).

X. ADVANTAGES

The proposed system offers several meaningful advantages over both manual screening processes and existing commercial ATS tools.

- 1) **Automation of Repetitive Tasks:** By handling resume ingestion, parsing, skill extraction, and matching without human involvement, the system frees HR professionals to focus on higher-value activities such as interviews and candidate relationship management.
- 2) **Faster Time-to-Shortlist:** Processing a single resume from upload to dashboard-ready results takes an average of under five seconds in the tested configuration, representing a reduction of several orders of magnitude compared with manual review.
- 3) **Personalised, Actionable Guidance:** Unlike systems that merely return a match percentage, the GenAI Skill Counsellor layer provides candidates with specific recommendations that they can act on immediately, positioning the platform as a career development tool as well as a recruitment aid.
- 4) **Horizontal Scalability:** The serverless Lambda architecture scales automatically with load. During periods of high application volume, additional execution environments are instantiated transparently, ensuring consistent response times without manual capacity planning.
- 5) **Reduced Cognitive Bias:** Standardised, score-based ranking reduces the influence of presentation factors such as resume aesthetic design, which can introduce unconscious bias in manual screening processes.

XI. FUTURE SCOPE

Several enhancements are planned to extend the capabilities and reach of the system.

Integration with External Job Portals: Connecting the platform with public job boards and professional networks would allow the system to proactively surface matching opportunities to candidates based on their profiles without requiring them to search manually.

AI-Driven Resume Improvement Suggestions: Beyond listing missing skills, future versions will offer line-level feedback on resume content, suggesting how candidates can better articulate existing experience to align with employer expectations.

Interview Preparation Module: By analysing the skill gap and the target role description, the system could generate likely interview questions and model answers, helping candidates prepare more thoroughly.

Real-Time Recruiter Analytics: Aggregated, anonymised data from the matching pipeline could be surfaced to HR teams as workforce analytics, revealing trends in candidate skill availability, emerging competency demands, and the effectiveness of job descriptions in attracting qualified applicants.

Multi-Language Support: Extending the NLP pipeline to handle resumes and job descriptions in languages other than English would make the platform accessible to a significantly broader global user base.

XII. CONCLUSIONS

This paper presented the design, implementation, and evaluation of an AI-powered resume skill matching and job recommendation system built on AWS serverless infrastructure. The system demonstrates that it is feasible to combine cloud-native scalability with state-of-the-art NLP and Generative AI techniques to deliver a recruitment support platform that benefits both job seekers and employers.

The proposed approach addresses the principal limitations of traditional resume screening by replacing subjective, time-consuming manual review with a transparent, data-driven pipeline. Candidates receive objective feedback on their position relative to job requirements along with personalised guidance on how to improve their competitiveness. HR teams gain a ranked candidate view that supports faster and more consistent shortlisting decisions.

Experimental evaluation on a set of 150 test resumes demonstrated a skill extraction precision of 91%, a recall of 87%, and a strong Spearman rank correlation of 0.84 between computed match scores and expert suitability ratings. These results confirm that the system performs well across diverse resume formats and job domains.

Future work will focus on integrating the platform with external job portals, introducing resume improvement suggestions, developing an interview preparation module, and expanding multi-language support to serve a global user base.

XIII. ACKNOWLEDGMENT

The authors wish to thank the faculty members of the Department of Computer Science and Engineering for their valuable guidance and support throughout this project. The authors also acknowledge the open-source NLP and cloud computing communities whose tools and documentation made this implementation possible.

REFERENCES

- [1] C.Fang,G.Huang,andB.Li,“ResumeKeywordExtractionandJob Matching Using Statistical Text Analysis,” in Proc. InternationalConference on Information Technology and Management Engineering,2018, pp. 112–117.
- [2] R. Maheshwari and A. Jain, “Automated Resume Screening Using Machine Learning Classifiers,” International Journal of Computer Applications, vol. 175, no. 14, pp. 24–29, Jun. 2020.
- [3] Y. Qin,Z.Hu,L.Liu,Y.Liu,andM.Sun,“Entity-DuetNeuralRanking:Understanding the Role of Knowledge Graph Semantics in NeuralInformation Retrieval,” Proc. 56th Annual Meeting of the Associationfor Computational Linguistics, 2018, pp. 2395–2405.
- [4] S. Shalaby, W. Zadrozny, and H. Jin, “Help Me Find a Job: A Graph-Based Approach for Job Recommendation at Scale,” in Proc. IEEEInternational Conference on Big Data, 2017, pp. 1544–1553.
- [5] D.BanerjeeandA.Roy,“Cloud-EnabledTalentAcquisition:AREviewof Modern ATS Platforms and Their Limitations,” Journal of HumanResource Technology, vol. 4, no. 2, pp. 45–58, 2021.
- [6] S.ChienandA.Mahadevan,“ServerlessComputingwithAWSLambda:DesignPatternsandBestPractices,”IEEECloudComputing,vol.6,no.1, pp. 30–39, Jan. 2019.
- [7] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: Pre-TrainingofDeepBidirectionalTransformersforLanguageUnderstanding,”Proc.NAACL-HLT, 2019, pp. 4171–4186.
- [8] AmazonWebServices,“AmazonDynamoDBDeveloper Guide,” AWS Documentation, [Online]. Available:<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/>
- [9] OpenAI, “GPT-4 Technical Report,” arXiv preprint arXiv:2303.08774,2023.
- [10] R.BharadwajandN.Tiwari,“SkillGapAnalysisinAutomatedRecruitment:ChallengesandOpportunities,”InternationalJournalofEmergingTechnologies in Learning, vol. 16, no. 4, pp. 78–91, 2021.
- [11] Vercel, “Next.js Documentation,” Vercel Inc., [Online]. Available:<https://nextjs.org/docs>
- [12] M.Zinkevich,M.Weimer,L.Li,andA.Smola,“ParallelizedStochasticGradientDescent,”AdvancesinNeuralInformationProcessingSystems,vol. 23, 2010.



10.22214/IJRASET



45.98



IMPACT FACTOR:
7.129



IMPACT FACTOR:
7.429



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24*7 Support on Whatsapp)