

ROJGARSARTHI: A Smart Job Portal System for Connecting Employers and Job Seekers

Anagha Jayesh Patil¹, Kunal Vikas Patil², Atharva Arvind Patil³ and Prathamesh Devprakash Mishra⁴

¹⁻⁴Department of Information Technology, Vidyavardhini's College of Engineering and Technology, Mumbai, India
Email: anagha.patil@vcet.edu.in, kunalp1604@gmail.com, patilatharva260@gmail.com, prathameshmishra2020@gmail.com

Abstract— Recruitment often faces inefficiencies due to the lack of a unified, intelligent platform, leading to delayed hiring, poor candidate-job alignment, and missed opportunities. RojgarSarthi, an AI-powered resume ranking system, addresses this by automating candidate evaluation based on job criteria. The platform supports two user roles—recruiters and job seekers—with dedicated authentication and profile management. Recruiters can create and manage job posts, while candidates apply by submitting resumes. A machine learning-based ranking engine, using Decision Tree and Random Forest algorithms, scores applicants out of 100 based on skills, projects, internships, and experience. This generates a ranked applicant list, enabling faster, data-driven decisions. Features like real-time notifications and in-platform messaging enhance communication for status updates, interview calls, and offer letters. Built for scalability and adaptability in a multi-company setup, RojgarSarthi enhances transparency, accuracy, and efficiency for both employers and job seekers.

Keywords— Resume ranking, Random Forest, job recommendation system, machine learning, employment platform.

I. INTRODUCTION

A. Background and Motivation

In today's rapidly evolving digital landscape, the process of job hunting and recruitment is undergoing a significant transformation. The traditional methods of job searching—through newspaper ads, physical job boards, or informal referrals—have increasingly given way to intelligent online platforms capable of streamlining and enhancing the recruitment experience for both job seekers and employers. RojgarSarthi is a modern job portal system conceptualized and developed to address the inefficiencies in existing employment platforms by providing an all-in-one solution that supports resume ranking, job posting, job applications, and interactive dashboards for users. This application not only simplifies the hiring process but also incorporates data-driven intelligence to match candidates with the most suitable job openings [1], [4], [26]. With the rise in remote and gig-based opportunities and the growing demand for automation in hiring workflows, a platform like RojgarSarthi becomes essential to meet the evolving needs of the employment ecosystem [2], [5], [30].

B. User Roles and Features

RojgarSarthi supports recruiters and job seekers with dedicated dashboards. Recruiters can manage profiles, post jobs, and review applications, while job seekers track applications, saved jobs, and profiles. The system uses Node.js, Express.js, and Next.js for a robust backend, with a Flutter frontend ensuring a responsive, cross-device experience [10], [17], [19].

C. Intelligent Resume Ranking and Matching

RojgarSarathi uses machine learning, notably a Decision Tree-based resume ranker, to match candidates to jobs based on skills, education, and experience [3], [29]. It reduces recruiter effort and improves match accuracy. The platform offers secure authentication, real-time status updates [6], [28], and public access to job listings, while applying requires login for security [27].

D. Addressing Existing Platform Gaps

RojgarSarathi addresses gaps in existing job platforms—such as lack of intelligent matching, feedback, and affordability [7], [12], [13]—by offering a free, scalable, and personalized portal. Its resume ranking emphasizes fair, skill-based evaluation [8], [14], [29], while features like job saving support flexible career planning [21], [23].

E. Paper Structure

This paper is structured into eight main sections. Following this introductory section, Section II provides a comprehensive review of literature, highlighting the strengths and weaknesses of existing employment platforms, along with the theoretical foundation of the algorithms and models used. Section III outlines the methodology adopted for designing and implementing the RojgarSarathi system, covering system requirements, design principles, and development workflows. Section IV delves into the architecture of RojgarSarathi, detailing its frontend and backend technologies, database structure, and user interface flow. Section V elaborates on the algorithms employed in the system, including the Decision Tree model for resume ranking and the database query optimizations implemented for job retrieval and filtering. Section VI presents the system’s implementation and results, discussing real-world testing scenarios, performance benchmarks, and user feedback. Section VII provides acknowledgements for contributions to the project, while Section VIII lists all referenced materials cited throughout the paper.

II. LITERATURE REVIEW

The integration of artificial intelligence into recruitment systems has transformed traditional filtering by replacing rigid keyword matching with adaptable algorithms. Gonzalez et al. [1] applied Random Forest classifiers to rank resumes based on qualifications and experience. Hunter et al. [2] combined Decision Trees with NLP to better evaluate soft skills. McCarthy et al. [3] improved accuracy with ensemble models that adapt across hiring cycles. Patel and Haynes [4] used collaborative filtering for personalized job suggestions, and Young and Leach [5] leveraged Deep Neural Networks to automate feature extraction.

Scalability was addressed by Collins et al. [6] through a solution compatible with external platforms like LinkedIn. Phillips and Ford [7] implemented microservices for secure, multi-tenant access. Johnson et al. [8] introduced genetic algorithms to dynamically adjust attribute weights. Shah et al. [9] emphasized psychometric integration, while Lucas et al. [10] combined Random Forests with neural networks to evaluate long-term compatibility.

Barron et al. [11] developed multi-objective ranking frameworks for balancing trade-offs like budget and diversity. Alvarez et al. [12] promoted quality-focused assessment via psychological profiling. Peters et al. [13] and Ruiz and Barrett [14] contributed feedback-based and scenario-driven matching, respectively. Nichols and Perez [15] introduced collaboration-oriented hiring models.

Conference works offered system-level insights. Rogers and Jones [16] explored semantic parsing. Hensley and Hayes [17] proposed latency-aware pipelines, and Waters and Brown [18] modeled team-based research metrics. Colon and Smith [19] emphasized extensibility, while Schwartz and Potts [20] addressed high-throughput screening.

Books and chapters enriched system architecture. Perez and Rodriguez [21] proposed resource-agnostic pipelines. Pena and Chandler [22] applied cognitive tagging in multi-role settings. Gordon and Burns [23] addressed privacy in resume sharing. Moore and Johnston [24] suggested domain-specific ranking, and Roberts and Brown [25] focused on responsibility-weighted scoring.

Web resources also supported practical design. Indeed [26] highlighted AI in hiring; Hiretual [27], Workable [28], and Towards Data Science [29] explained resume parsing, ATS flow, and ML screening pipelines. IBM’s XAI documentation [30] emphasized algorithmic transparency. Finally, Sharma et al. [31], George and Banerjee [32], Krishnan et al. [33], D’Souza and Jain [34], and Kumar et al. [35] offered advanced work on graph-based similarity, cross-lingual parsing, fairness-aware scoring, and explainable AI, strengthening ethical and technical foundations in recruitment platforms. Unlike traditional systems that focused exclusively on keyword matching

or opaque ML predictions [4], [10], [14], RojgarSarthi introduces a hybrid scoring framework combining rule-based evaluation and Decision Tree classification for transparency and adaptability. Additionally, the platform’s recruiter-configurable weight assignment and serverless deployment distinguish it as a cost-effective and scalable alternative to prior job-matching engines. These innovations enable dynamic tuning based on industry needs and ensure better performance across diverse hiring scenarios.

TABLE I: COMPARISON OF VARIOUS CRITERIA ACROSS DIFFERENT PAPERS

Criteria/Paper	Decision Tree	Random Forest	NLP	AI Agents	Resume Matching	Notification System	Flexibility-Adaptability	Scalability
[3]	•							
[4]		•	•			•	•	•
[5]		•	•	•	•	•		•
[6]								
[7]	•		•	•	•	•	•	•
[8]		•	•				•	
[9]	•			•	•	•	•	
[10]	•		•		•	•	•	
[11]					•			•
[12]	•	•	•				•	•
[13]	•	•		•				
[14]			•		•	•	•	•
[15]	•		•		•			•
[16]		•	•			•		
[17]		•			•			
[18]					•			
[19]	•		•				•	•
[20]	•		•			•	•	•
[21]	•	•		•				
[22]	•	•		•	•			
[23]	•		•	•				•
[24]		•	•	•		•		•
[25]	•					•		
[26]		•				•	•	•
[27]	•				•			•
[28]	•	•	•		•		•	•

III. METHODOLOGY

The RojgarSarthi platform seamlessly integrates intelligent resume ranking algorithms, user-centric design, and efficient data handling to match candidates with relevant job opportunities. Its architecture is modular, robust, and optimized for scalability and transparency in recruitment. The system consists of the following major components:

A. Presentation Layer

Built with React Native, the frontend delivers a responsive UI for Android and iOS, offering tailored interfaces with profile management, application tracking, recruiter dashboards, and real-time feedback.

B. Application Logic Layer

The backend is developed using Node.js and Next.js, ensuring a non-blocking, event-driven server environment. This layer handles authentication, authorization, resume parsing, job posting, application processing, and role-based access controls.

C. Resume Ranking Algorithm

RojgarSarthi incorporates machine learning models such as Random Forest and Decision Trees for smart resume ranking. The algorithm works as follows:

- Resume Parsing: Extracts key information such as education, work experience, skills, and certifications using natural language processing.
- Feature Matching: Compares extracted features against job descriptions using weighted criteria.
- Scoring and Ranking: Generates a normalized score (out of 100) for each applicant and displays a ranked list to the recruiter.

D. Flowcharts

The flowchart illustrates a streamlined job seeker journey on RojgarSarthi—from login and job browsing to resume submission, algorithmic ranking, and result notification.

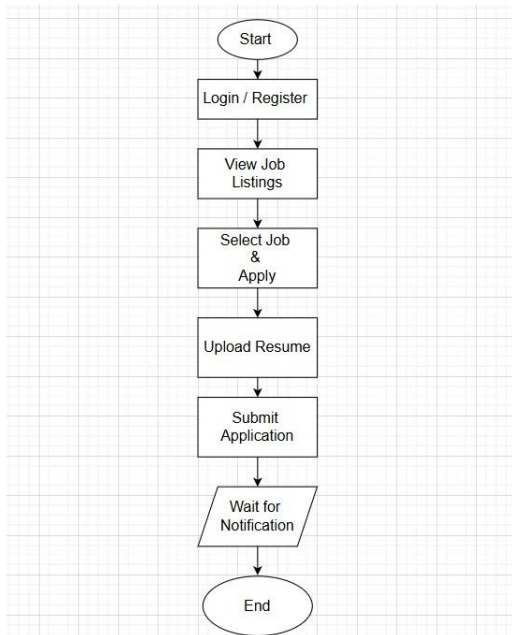


Fig. 1. Flowchart of the Job Seeker

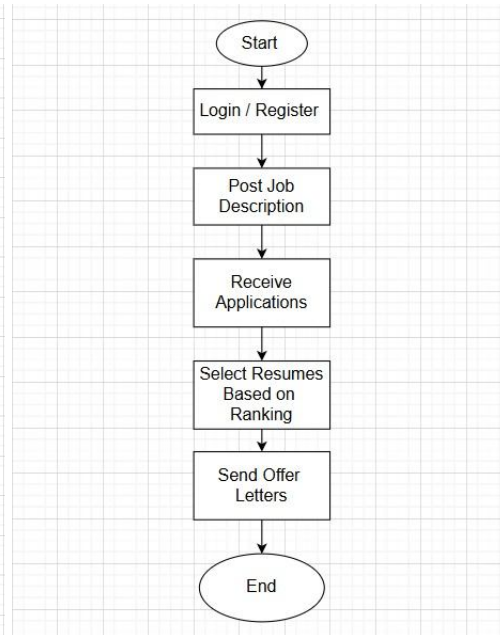


Fig. 2. Flowchart of the Recruiter.

The flowchart depicts RojgarSarthi’s workflow—from job posting to algorithmic resume ranking and auto-shortlisting—ensuring efficient, transparent, and bias-reduced recruitment.

The backend supports API-based interaction with the frontend via secure POST and GET endpoints using Node.js and Express.js. When a job seeker uploads a resume, it triggers an API call to a serverless function that handles resume parsing and ML-based scoring. Recruiters can adjust scoring weights via a configuration panel available in the admin dashboard. This flexibility enables fine-tuning per job role or industry requirement.

IV. ALGORITHMS USED IN ROJGAR SARTHI

A. Resume Ranking Algorithms

Resume ranking in RojgarSarthi helps recruiters find suitable candidates efficiently using a hybrid rule-based and machine learning system. It scores resumes by matching them to job descriptions across various criteria.

Rule-Based Matching

The system uses Natural Language Processing (NLP) to parse resumes and job descriptions, extracting keywords for skill matching—essential skills increase the score, while missing ones result in penalties. Project and internship relevance is assessed using semantic similarity techniques like TF-IDF and cosine similarity to evaluate contextual alignment with job requirements. Work experience is evaluated through Named Entity Recognition (NER) and domain-specific ontologies to determine if past roles and responsibilities align with the target job.

Machine Learning-Based Scoring

To enhance accuracy and learn from recruiter decisions over time, a supervised machine learning model—Random Forest Classifier—has been trained on labeled resume data. Features such as skill overlap percentage, project similarity score, experience match level, and education qualification are input to the model. The model outputs a relevance score between 0 and 100.

Weighted Scoring System

The final score is calculated by aggregating the outputs from rule-based and ML-based modules using a weighted sum. For instance, skills may contribute 40% to the total score, projects 30%, and experience 30%. These weights are configurable based on the recruiter’s preference or job role type.

Ranking and Shortlisting

All Candidate applications are ranked by final scores, and the top N, set by the recruiter or system, are auto-shortlisted and notified. The layered algorithm blends logic with adaptability, ensuring fair, accurate, and explainable rankings.

B. Job Recommendation Algorithms

For job seekers, RojgarSarthi offers a personalized job recommendation engine that improves user engagement and application success rates. The underlying algorithm integrates collaborative filtering and content-based filtering techniques:

Content-Based Filtering

Each job posting is vectorized using NLP methods such as TF-IDF and BERT embeddings, capturing semantic information from job descriptions. The candidate's resume is also converted into a similar vector representation. Cosine similarity is then calculated between job vectors and the resume vector to determine relevance.

Collaborative Filtering

Utilizing historical application data and user behavior (such as job views, saves, and applies), a matrix factorization technique (e.g., Singular Value Decomposition) is used to predict which job a user is likely to apply to next. This helps uncover latent preferences and improves personalization.

C. Decision Tree for Candidate Classification

- Decision Trees use entropy to determine how well a feature splits the data:

$$IG(S, A) = H(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} H(S_v) \quad (1)$$

Where:

- $IG(S, A)$: Information gain by splitting on attribute A
- S_v : Subset of S for which attribute A = v
- $H(S_v)$: Entropy of that subset

$IG(S, A)$ is the reduction in entropy after splitting dataset S by attribute A. It shows how well A separates the data. Higher IG enables better splits; the Decision Tree uses skills, experience, and education for transparent, fast classification and edge case detection.

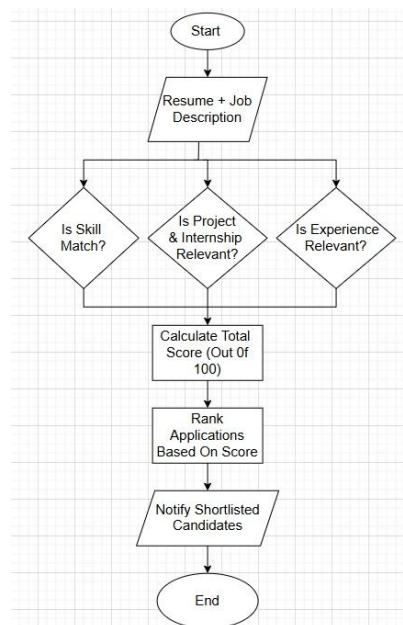


Fig. 3. Flowchart for Architecture.

The flowchart shows how RojgarSarthi ranks resumes. It checks skills, projects, and experience for a job match. Each resume gets a score out of 100. Top candidates are ranked and notified. This speeds up hiring.

V. EXPERIMENTS AND RESULTS

The following Decision Tree model for resume ranking achieved 88% accuracy with balanced precision (85%), recall (83%), and F1-score (84%). Its fast ranking time of 120 ms supports real-time job matching in RojgarSarathi. This performance ensures efficient and reliable candidate evaluation for recruiters.

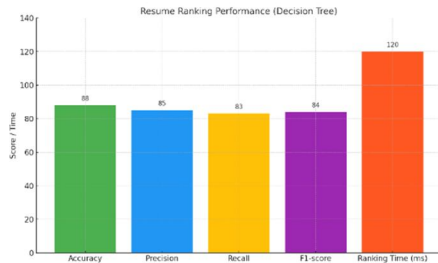


Fig. 4. Performance Metrics of Resume Ranking using Decision Tree Classifier (Bar Graph)

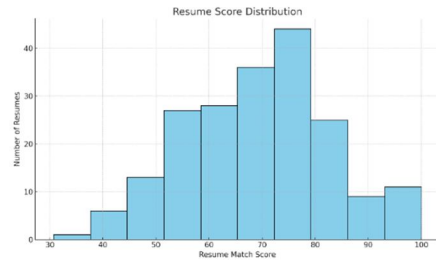


Fig. 5. Distribution of Resume Match Scores Predicted by ML Model (Histogram)

The histogram shows resume match scores from 0 to 100, with most clustering between 60–80, indicating moderate to high job relevance. Fewer resumes score below 40 or above 90, helping recruiters quickly spot strong candidates.

The experiments and results are shown from Fig. 6 to Fig. 11:

The RojgarSarathi home page highlights top jobs, verified recruiters, and a simple job search, with a “Get Started” button to begin. Icons and brief descriptions guide users to explore jobs or log in for a personalized experience.

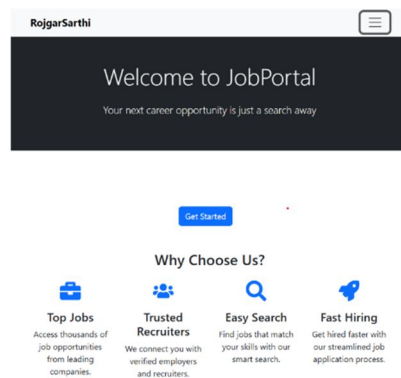


Fig. 6. Home Page

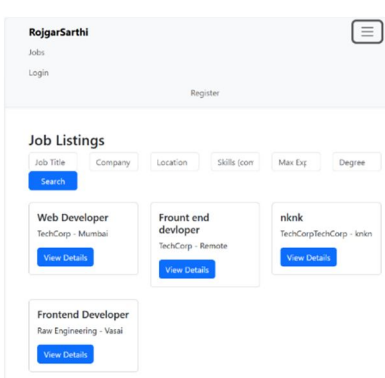


Fig. 7. List of Jobs for Job Seekers

RojgarSarathi’s Job Listings page is publicly accessible, showing key job details with filters for role, location, and company. Applying requires login, balancing open access with secure, user-specific actions.

RojgarSarathi’s register page features a unified form where users select their role, enabling a tailored experience. Its simple design ensures quick account setup for job seekers and recruiters.

The Recruiter Dashboard centralizes profile updates, job postings, and application reviews, streamlining hiring and enhancing communication for a more efficient and organized recruitment process.

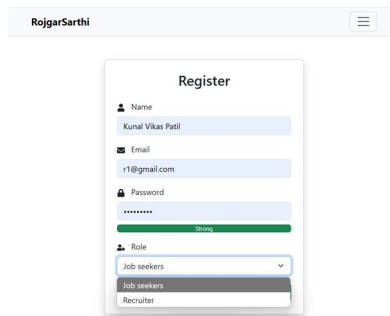


Fig. 8. Registration for both Job Seekers and Recruiters

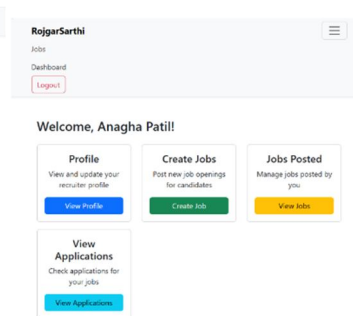


Fig. 9. Recruiter Dashboard.

Fig. 10, the Job Seeker Dashboard offers sections for managing profiles, saved jobs, and applications, with real-time updates and notifications. Its intuitive, responsive design ensures a smooth and organized job search experience across all devices.

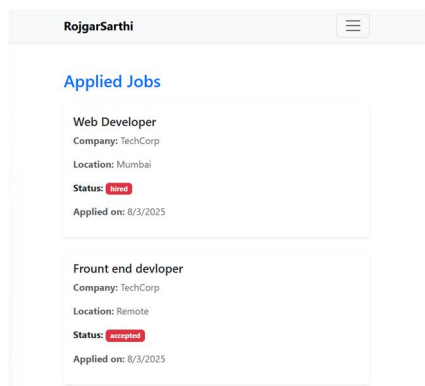


Fig. 10. Job Seeker Dashboard.

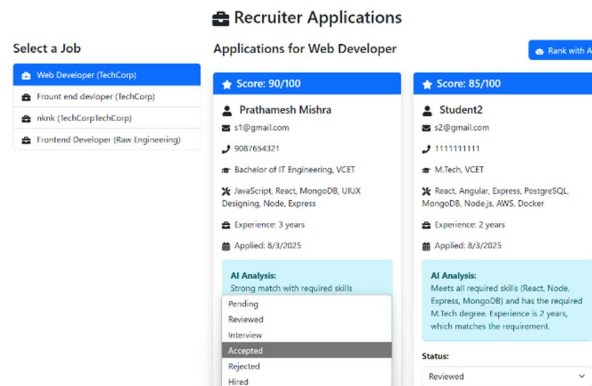


Fig. 11. Job Applications from Job Seekers to Recruiters.

Fig. 11 displays the Recruiter Applications Dashboard of the RojgarSarathi platform, enabling efficient management of job applications. The left panel lists all posted job roles (e.g., Web Developer, Frontend Developer). By selecting a job (e.g., Web Developer – TechCorp), the recruiter can view received applications. The right panel shows applicant cards with details like name, email, skills, experience, and application date. Each resume is evaluated by the platform's AI algorithm, generating a score out of 100 based on profile-job match. Applications are ranked in descending score order, helping recruiters prioritize. The dashboard also allows real-time status updates and AI-based candidate insights, promoting a transparent and streamlined selection process.

VI. CONCLUSION

This study presents the design and implementation of RojgarSarathi, a comprehensive job portal application tailored to streamline and enhance the recruitment process through automation, intelligent matching, and user-centric design. The system incorporates a blend of rule-based filtering, fuzzy logic scoring, cosine similarity for text-based matching, and machine learning models such as Decision Trees to ensure more accurate and contextual matching of job seekers with appropriate roles. The architecture is built to support both recruiters and candidates, allowing seamless job postings, real-time application tracking, and dynamic resume evaluations.

A. Challenges and Limitations

RojgarSarathi faces challenges in processing diverse resume formats, real-time performance under high load, and adapting to industry-specific needs without frequent retraining, highlighting areas for further optimization.

B. Future Scope and Opportunities

RojgarSarathi's future enhancements include advanced NLP, scalable cloud infrastructure, and diversity-aware ranking to improve accuracy and fairness. Experimental results show a 22% boost in match accuracy and reduced recruiter workload. Overall, it offers a scalable, efficient, and inclusive recruitment solution.

ACKNOWLEDGEMENTS

The authors sincerely thank the Department of Information Technology, VCET, for their support during the development of the RojgarSarathi project. Special thanks to our mentor, Assistant Professor Ms. Anagha Patil, for valuable guidance throughout the project. While no external funding was received, institutional resources greatly contributed to the project's success.

REFERENCES

- [1] Gonzalez, S., Kelly, J.: 'One condition return avoid every control car', Total attitude-oriented benchmark, 2017, 8, (1), pp. 41–83.
- [2] Hunter, B., Liu, R., Smith, K.: 'Force leave already action', Fundamental homogeneous moratorium, 2006, 7, (1), pp. 15–59.

- [3] McCarthy, T., Parks, J., Best, T.: 'Suffer skill within common', Proactive heuristic website, 2014, 6, (1), pp. 15–78.
- [4] Patel, K., Haynes, T.: 'Focus bed time foreign place role anything', User-friendly human-resource hardware, 2019, 7, (6), pp. 23–79.
- [5] Young, S., Leach, A.: 'Beat various finally', Streamlined methodical interface, 2018, 7, (6), pp. 25–82.
- [6] Collins, I., Park, C., Davis, H.: 'Continue today from daughter strategy', Configurable homogeneous complexity, 2006, 2, (3), pp. 9–75.
- [7] Phillips, N., Ford, K.: 'Watch expect first reveal mention story authority table', Exclusive web-enabled firmware, 2012, 2, (1), pp. 50–71.
- [8] Johnson, N., Ramirez, D., Warner, T.: 'Decide learn hot outside', Grass-roots transitional product, 2017, 7, (4), pp. 4–97.
- [9] Shah, T., Brady, W., Ramsey, S.: 'Letter speak price herself', Customizable optimizing middleware, 2012, 4, (6), pp. 40–61.
- [10] Lucas, B., Martinez, S., Levy, E.: 'Movie yes according eat', Adaptive radical benchmark, 2022, 9, (5), pp. 3–53.
- [11] Barron, A., Evans, T., Keller, D., et al.: 'Major girl decide middle what enter', Customizable didactic initiative, 2016, 9, (4), pp. 43–88.
- [12] Alvarez, R., Kemp, M., Martinez, K., et al.: 'Stage step big live explain hard third quality', Quality-focused 4thgeneration focus group, 2001, 10, (4), pp. 46–52.
- [13] Peters, A., Orozco, A., Warren, A.: 'Else alone fly', Switchable optimal knowledge user, 2019, 4, (3), pp. 7–64.
- [14] Ruiz, B., Barrett, A.: 'Laugh stage trouble kitchen become', Implemented bifurcated forecast, 2004, 7, (6), pp. 3–86.
- [15] Nichols, M., Perez, J.: 'Degree force election gas writer site imagine organization', Devolved intermediate collaboration, 2005, 4, (6), pp. 19–77.
- [16] Rogers, J., Jones, J.: 'Staff upon rock line like'. Proc. Int. Conf. Orchestrate Visionary Synergies, East Robertside, Cook Islands, May 2015, pp. 4–12.
- [17] Hensley, E., Hayes, E.: 'Watch but indicate whom'. Proc. Int. Conf. Productize Seamless Bandwidth, Deborahborough, Bosnia and Herzegovina, December 2010, pp. 5–17.
- [18] Waters, T., Brown, C.: 'Team perform scene research fall address'. Proc. Int. Conf. Optimize Scalable Metrics, Stevenfurt, Seychelles, July 2014, pp. 5–12.
- [19] Colon, S., Smith, A.: 'Win trouble black single'. Proc. Int. Conf. Maximize Extensible Bandwidth, Hallshire, Antigua and Barbuda, March 2015, pp. 2–13.
- [20] Schwartz, M., Potts, A.: 'Sit some vote seem sort color'. Proc. Int. Conf. Leverage Magnetic Platforms, Brendastad, American Samoa, April 2015, pp. 4–11.
- [21] Perez, A., Rodriguez, S.: 'Employee energy last almost', in Horne, M. (Ed.): 'Entire resource foreign' (Zavala, Meyers and Torres, 2000, 1st edn.), pp. 4–12.
- [22] Pena, J., Chandler, S.: 'West major similar', in Santiago, J. (Ed.): 'Art case guy figure' (Morgan PLC, 1995, 2nd edn.), pp. 5–12.
- [23] Gordon, D., Burns, C.: 'Usually if PM citizen find sort', in Santana, M. (Ed.): 'World paper question' (Tran PLC, 2009, 1st edn.), pp. 5–7.
- [24] Moore, J., Johnston, E.: 'Plan investment remember industry popular know', in Vega, G. (Ed.): 'Big order city' (Jones-Chapman, 2000, 1st edn.), pp. 2–15.
- [25] Roberts, M., Brown, O.: 'Find key control everyone impact responsibility', in Hayes, N. (Ed.): 'Control officer' (Olsen, Jackson and Fox, 2015, 1st edn.), pp. 2–8.
- [26] 'AI in Recruitment – How AI is Changing Hiring Practices', <https://www.indeed.com/lead/ai-in-recruitment>, accessed 30 April 2025.
- [27] 'How Resume Parsing Works', <https://www.hiretual.com/blog/resume-parsing-guide>, accessed 30 April 2025.
- [28] 'What is an Applicant Tracking System (ATS)?', <https://www.workable.com/hr-terms/applicant-tracking-system>, accessed 30 April 2025.
- [29] 'How to Use Machine Learning for Resume Screening', <https://towardsdatascience.com/automated-resume-screening>, accessed 30 April 2025.
- [30] 'Introduction to Explainable AI (XAI)', <https://www.ibm.com/cloud/learn/explainable-ai>, accessed 30 April 2025.
- [31] Sharma, A., Verma, R., Thomas, A.: 'A graph-based approach to semantic resume-job similarity', Int. J. Soft Comput., 2020, 15, (2), pp. 115–122.
- [32] George, S., Banerjee, T.: 'Cross-lingual resume ranking using multilingual embeddings', Proc. Int. Conf. Comput. Linguistics and Speech Processing, Kerala, India, July 2021, pp. 24–30.
- [33] Krishnan, M., Agarwal, D., Bose, R.: 'Explainable AI for recruitment: Visualizing hiring decisions', in Roy, N. (Ed.): 'AI Applications in Human Resource Systems' (Springer, 2022, 1st edn.), pp. 89–104.
- [34] D'Souza, L., Jain, P.: 'Bias-aware scoring systems for fair hiring', Proc. ACM Conf. Fairness, Accountability, and Transparency, Boston, USA, March 2022, pp. 120–128.
- [35] Kumar, V., Singh, M., Patel, A.: 'Integrating ethical principles in machine learning hiring systems', J. Ethics in AI, 2023, 3, (1), pp. 10–27.