# Job Match an Android Recruitment Platform Integrating OCR and Semantic Matching

**Fatin Nur Hannah Binti Muhammad Ashraf, Elaiyaraja Krishnan, Muhamad Amirul Asyraf Rahmat**

**[1]School of Computing, Faculty of Computing and Engineering, Quest International University, Malaysia**

**DOI: https://doi.org/10.47772/IJRISS.2026.101000601**

**Received: 25 October 2026; Accepted: 30 October 2026; Published: 11 November 2026**

## ABSTRACT

Digital recruitment requires platforms to interpret information containing resumes and other supporting documents related to candidate capabilities to job requirements. The study showcased an Android recruitment prototype called JobMatch that integrates optical character recognition (OCR), user-reviewed skill extraction, and semantic matching to support job discovery and recruiter-side candidate ranking, using Tesseract.js for document text extraction and Sentence-BERT (SBERT) embeddings for semantic comparison. This article demonstrates a descriptive pilot evaluation involving 14 job seekers and 2 recruiters. The project reported overall satisfaction of 4.56 out of 5 and perceived matching relevance of 4.50 out of 5, while preliminary matching tests reported mean Precision@K of 0.750 and mean nDCG@K of 1.000 using illustrative test labels. The OCR integration test returned 18 skill suggestions for user review; its engine confidence score is not an objective measure of OCR accuracy. The findings support the feasibility of integrating these functions in a mobile recruitment workflow, but do not establish independently validated matching effectiveness or document-recognition accuracy. The practical contribution is the inclusion of a user-confirmation checkpoint before document-derived skills are used in matching.

**Keywords:** recruitment platform; Android application; optical character recognition; Sentence-BERT; semantic similarity; candidate ranking

## INTRODUCTION

Recruitment platform must accurately connect the description of candidate capabilities with employers' requirements. Provided documents and information, such as resumes, certificates, profiles and job advertisements, may showcase related information in different forms. For example, a job seeker may describe "Android development experience ", while a recruiter may ask for "mobile application development", rendering exact-term comparison failure to recognise this relationship. Additionally, relevant information may remain embedded in uploaded documents rather than being available as structured profile data. These issues provoke a workflow that combining document text extraction with semantic comparison.

JobMatch Mobile Application was developed as a prototype to address these integration challenges, providing jobseeker functions for profile management, document submission, job discovery, applications and application tracking, while recruiters can access recruitment management functions, including job listing management, applicant review, and candidate matching information where supported by the implemented build. The integration of optical character recognition (OCR) and embedding-based semantic comparison support job recommendation and candidate ranking. These output are intended to assist decision-making rather than replace human hiring decision.

The project had three implementation objectives: (1) to develop an integrated mobile recruitment platform with document-processing capabilities; (2) to implement SBERT-based semantic matching between candidate information and job requirements; and (3) to provide job recommendations and recruiter-side candidate ranking. The study addressed three research questions: RQ1: How can OCR support the extraction of

information from recruitment documents in JobMatch? RQ2: How can SBERT-based semantic comparison be used to relate candidate information to job requirements? RQ3: What preliminary evidence do functional tests and pilot-user feedback provide about the operation and perceived usefulness of the JobMatch workflows? RQ1 was examined through OCR and skill-extraction scenarios; RQ2 through the semantic-matching test cases and ranking metrics; and RQ3 through functional and integration tests and the descriptive questionnaire results. The available evidence does not establish comprehensive OCR accuracy or real-world recruitment-ranking effectiveness.

The contribution is toward the practical integration of already established technologies rather than the development of a new language model or OCR algorithm. A development in a distinctive workflow feature is the review of OCR-derived skill suggestions, allowing users to confirm or reject suggested skills before confirmed information is incorporated into candidate profiles and used for matching. This article synthesises the background, methodology, implementation, evaluation, and conclusions presented in the underlying FYP report. (Muhammad Ashraf, 2026).

## Related Work and Technical Foundations

### Recruitment and Job Recommendation Systems

Existing research explores different approaches, including skill-based extraction, job recommendation and vacancy ranking. For example, Gugnani and Misra (2020) highlight the role of candidate skills extraction and job recommendation and vacancy ranking by examining skill extraction and job recommendation. Another example, Vanetik and Kogan (2023) demonstrate another approach to organising and prioritising employment opportunities by investigating job vacancy ranking. The Recruitment and job recommendation systems aim to improve the process of connecting job seekers with suitable employment opportunities. These studies address an important part of the recruitment process, but with some questions regarding skill extraction and vacancy ranking, they represent different tasks and should not be treated as evidence of a complete recruitment workflow.

### Keyword-Based and Semantic Matching

Keyword-based matching identifies candidates through shared words between resumes and job descriptions. However, it may overlook relevant candidates when similar skills are expressed using different terms. Therefore, Reimers and Gurevych (2019) introduced Sentence-BERT (SBERT), which generates sentence embeddings that can be compared using cosine similarity. JobMatch uses SBERT to identify semantic relationships between candidate information and job requirements. It also combines semantic similarity (80%) with explicit skill coverage (20%) to calculate the final matching score.

### Resume and Skill Extraction

Resumes and certificates contain useful information, such as qualifications, work experience, and skills. However, this information may be stored in unstructured documents. Optical Character Recognition (OCR) helps extract text from scanned documents. Smith (2007) describes the Tesseract OCR engine, which provides the foundation for the Tesseract.js component used in JobMatch. Extracted text is processed to identify potential candidate skills. However, OCR may produce errors, and extracted information should not automatically be treated as verified. JobMatch therefore allows job seekers to confirm or reject suggested skills before they are included in candidate profiles and used for matching.

### Human Review in AI-Assisted Recruitment

AI can support recruitment by processing candidate information and ranking potential matches. However, automated results may contain errors or reflect bias. Raghavan et al. (2020) highlight the importance of evaluating fairness and bias in algorithmic hiring systems. Therefore, JobMatch incorporates human review

by allowing job seekers to validate extracted skills and recruiters to review matching results. The system is designed to support recruitment decisions rather than replace human judgement.

# METHODOLOGY

### Development approach

The FYP follows an Agile methodology approach over the planned period from **20 April 2026 to 9 October 2026,** developing and dividing the implementation into small, manageable stages. Each stage focuses on implementing, integrating, and testing selected features before moving on to the next stage. The cycle for each feature starts with requirements and planning, design and development, testing and integration, and evaluation and improvement. Requirements were organised around the main activities of jobseekers and recruiters.

The development process combined interface planning with the incremental implementation of the Android application and its supporting services. Testing was conducted to verify each feature, while integration testing assessed communication between the application, backend, database, OCR component, and semantic matching service. Any issues identified during testing were addressed through refinement and further testing.

Following development activities focus on job listings, job posting, job search, applications, applicant management and document uploading. The OCR uses Tesseract.js to extract text from resumes and certificates, and an added skill confirmation workflow was developed to allow job seekers to review, confirm or reject extracted skill suggestions before being used in SBERT. Separately, the Python FastAPI service was developed using Sentence Transformers (SBERT) to generate semantic embeddings and calculate cosine similarity scores. These scores were subsequently integrated into the recommendation and candidate-ranking functions.

The Testing covered the main application functions and communication between the Android application, backend API, database, OCR component and semantic matching service. OCR extraction quality and matching behaviour were evaluated using prepared resumes and job descriptions.

Pilot testing was conducted with job seekers and recruiters to gather feedback on the application's usability and functionality. Issues identified during testing and feedback collection informed subsequent fixes and refinements, followed by further testing of the affected functions.

The final stages involved updating the project documentation and supporting evidence, preparing the Innovation Day poster and demonstration materials, and conducting final verification before submission (Muhammad Ashraf, 2026).

The study is presented as a prototype development and descriptive pilot evaluation. The evidence consists of the system architecture, implemented workflow descriptions, reported test scenarios, and questionnaire means in the final report. No new experiment was performed for this article. Source-code behaviour, raw questionnaire responses, and ground-truth matching judgements were not independently reanalysed.

### Participants and evaluation procedure

The pilot evaluation involved 16 participants, consisting of 14 job seekers and 2 recruiters or human resource (HR) respondents. Participants provided background information, including their user role, age group and previous experience using online recruitment platforms. The recorded age groups included 18–20, 21–25, and 31–40 years. This information provided a basic overview of the participants' backgrounds and their familiarity with digital recruitment services.

The evaluation was conducted using a Google Forms questionnaire after participants had tested the relevant JobMatch features. The evaluation covered the main functions available to each user role. Job seekers evaluated registration and profile setup, job searching and browsing, application tracking, AI-generated match

scores, recommendation relevance, interface navigation and the overall usefulness of the application for job discovery. Recruiters evaluated company account setup, job listing management, applicant information, AI matching scores, candidate ranking, and applicant management and shortlisting.

Participants rated their experiences using a five-point Likert scale, ranging from 1 (strongly disagree) to 5 (strongly agree).

The questionnaire also included questions on perceived AI matching accuracy, recommendation relevance and OCR extraction quality, including whether participants encountered incorrect or missing information in extracted text. Open-ended questions allowed participants to report system errors, identify useful features and suggest improvements.

The responses were analysed descriptively using item-level mean scores to summarise participants' perceptions of the system. Open-ended responses were reviewed to identify recurring feedback and possible areas for improvement. The evaluation involved a relatively small sample, particularly the two recruiter respondents. Therefore, the findings are interpreted as preliminary user feedback rather than statistically generalizable evidence of user acceptance or system effectiveness.

**Testing strategy**

Functional testing compare expected and observed behaviours for account management, recruitment operations, document processing, recommendations, and application tracking. Integration tests followed requests and information between the mobile application, backend, database, OCR process, and semantic service.

Separate OCR, skill-extraction, and semantic-matching checks examined whether the relevant components operated on the supplied test inputs. These checks establish reported scenario execution and preliminary component behaviour, rather than exhaustive coverage or independently measured algorithmic accuracy.

The reported results includes 14 functional scenarios and 12 integration scenarios, all recorded as Pass. Five OCR scenarios were also recorded as Pass, while five semantic-matching scenarios were recorded as Successful. The results were recorded according to whether the tested operations produced their expected outcomes, and a test considered passed when the observed behaviour satisfied the expected results for tested scenario.

These results shows the execution of selected test scenarios and the preliminary operational behaviour of the prototype and does not establish exhaustive system coverage, independently measured OCR accuracy, or validate real world recommendation effectiveness. Additional technical checks involving OCR, skill extraction and semantic matching are presented separately in Section 5.4.

**System Architecture and Implementation**

**Layered architecture**

The Android client was implemented using Java and XML, while Retrofit connect the client to a Node.js and Express REST backend, which then coordinates recruitment operations and relational persistence in MySQL. A separate Python FastAPI service performs semantic matching using Sentence Transformers.

This allows the separation of mobile interface, business logic, data storage, and language-processing components to have distinct tasks. The overall architecture of the JobMatch system, showing the communication between the Android application, backend API, database, OCR component, and semantic-matching service, is illustrated in Figure 1.

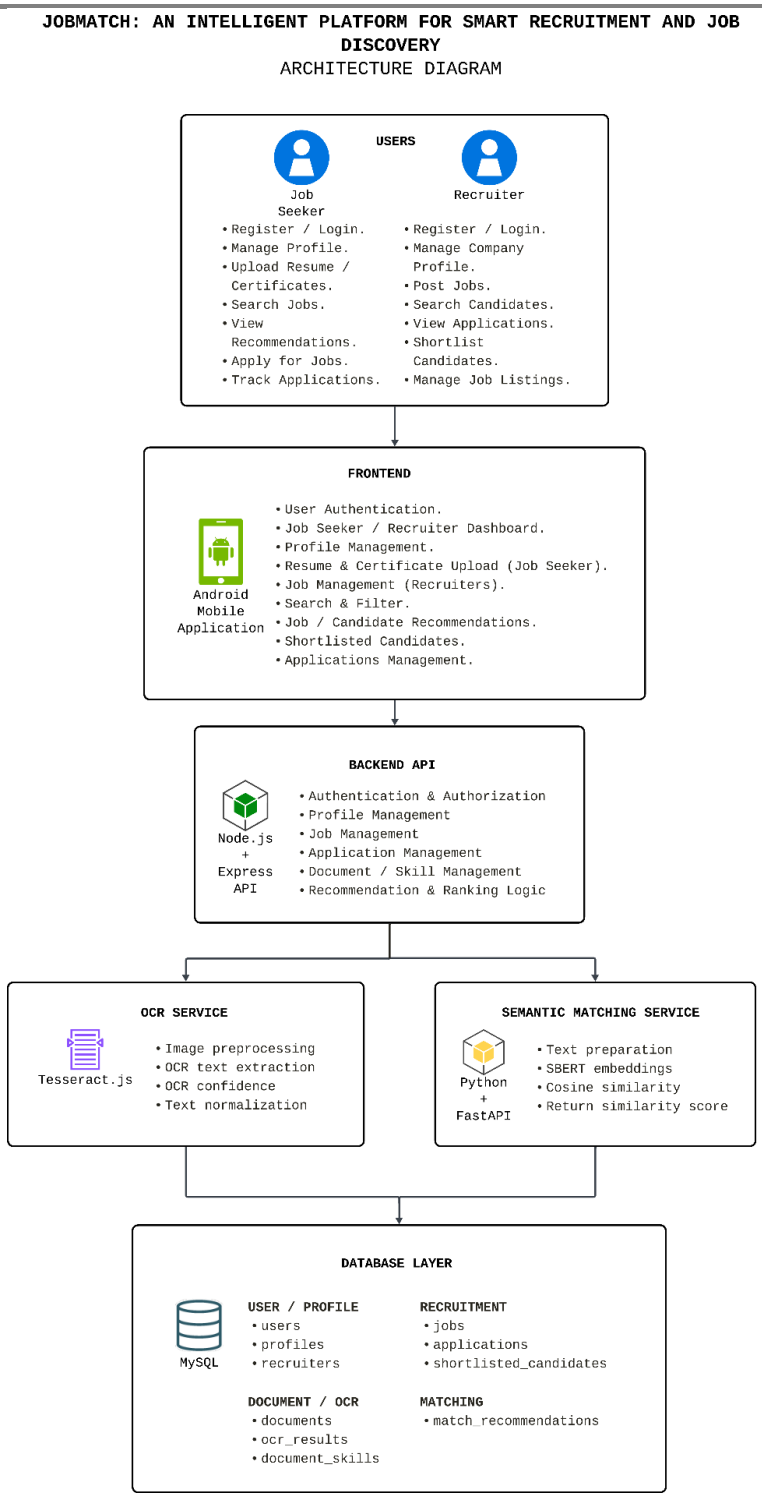


**Figure 1. JobMatch System Architecture**

| Component | Implemented role |
|---|---|
| Android Java and XML | Job-seeker and recruiter interfaces |
| Retrofit | HTTP requests between Android and the backend |
| Node.js and Express | REST endpoints and recruitment workflow coordination |

| | |
|---|---|
| MySQL | Accounts, profiles, jobs, documents, applications and scores |
| Tesseract.js | Document text extraction |
| Python FastAPI and SBERT | Embedding generation and semantic comparison |

**Document processing and skill review**

A job seeker uploads a resume, certificate, or supporting document, while the backend initiates OCR and stores extracted content in an ocr_results record. The report shows storage of text and associated information such as name, issuer, date, and recognition confidence where available. Potential skills are stored separately in document skills. Each suggestion can be pending, confirmed, or rejected. Additionally, only confirmed document-derived skills contribute to candidate matching information. Skill identification uses a predefined skill vocabulary and rule-based text processing to detect relevant technical skills and qualifications in the extracted content. The extraction process includes text normalisation, skill-term matching and duplicate prevention. It also handles selected equivalent terms by mapping recognised variations to standard skill labels. For example, "Entity Relationship Diagram" is mapped to "ERD". The extraction rules also distinguish similar but different terms, such as Java and JavaScript, and GitHub and Git, to reduce incorrect matches.

Extracted skill suggestions are initially assigned a pending status, and Job seekers can review each suggestion and either confirm or reject it. Only confirmed document-derived skills are incorporated into candidate matching information. OCR recognition confidence is retained where available as information about the text-extraction process. However, it is not treated as a direct measure of skill correctness since it does not establish a separate numerical confidence threshold for accepting skill suggestions. The overall OCR and skill-confirmation process as illustrated in Figure 2.

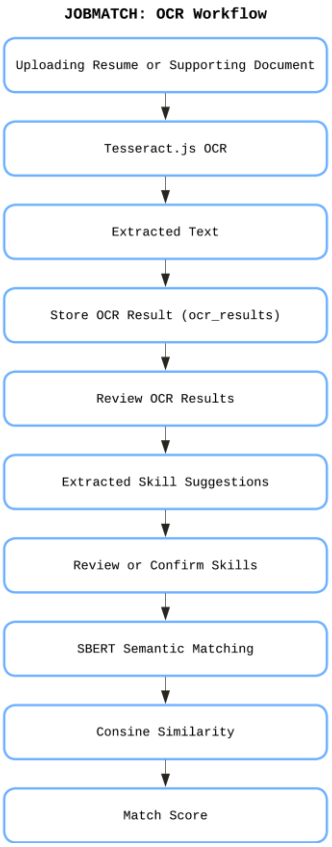


**Figure 2. OCR and Skill-Confirmation Workflow**

The review phase establishes a practical checkpoint between automated recognition and profile enhancement. The workflow enhances user authority over retrieved information without confirming the accuracy of each foundational qualification. This is because a user has the option to dismiss a misidentified or unrelated skill prior to it affecting suggestions. Nonetheless, confirmation does not serve as an autonomous verification of skill or document validity.

The OCR process compared enhanced-image and threshold-image preprocessing, with both approaches reporting a confidence score of 92; the enhanced image was selected for subsequent extraction. The test produced 18 suggested skills, all of which remained pending user review. However, the implementation documentation does not fully specify the supported file extensions, maximum file size, PDF page limits, OCR language configuration, preprocessing parameters or failure-handling procedure for unsupported, corrupted or unreadable documents. These details should be documented before replication. Extraction success in the listed examples is therefore described at the workflow level without assuming universal support for every document layout or file type.

**Semantic comparison and recommendations**

JobMatch uses the Sentence Transformers model sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2 to generate semantic embeddings for candidate and vacancy information. The model is loaded through the Sentence Transformers library and runs within a separate Python FastAPI service.

Candidate text is assembled from available skills, education, experience, profile information, and confirmed document-derived skills. Vacancy text incorporates the job title, description, skills, qualifications, and other requirements. The semantic-matching process shows how candidate information and job requirements are represent as sentence embeddings and compared using cosine similarity to support job recommendation and recruiter-side candidate ranking, as illustrated in Figure 3. The FastAPI service encodes these representations using SBERT and compares the resulting vectors. For nonzero candidate vector c and vacancy vector j, cosine similarity is defined as:

$$s(c, j) = \Sigma_i(c_i j_i) / [\sqrt{\Sigma_i(c_i^2)} \times \sqrt{\Sigma_i(j_i^2)}]$$

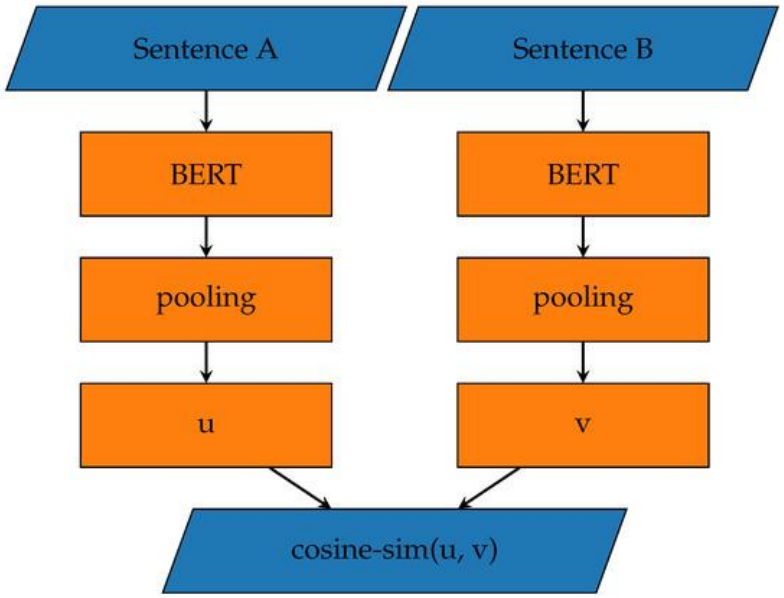


**Figure 3. SBERT-Based Semantic Matching Process**

Note. From An Intelligent Conversational Agent for the Legal Domain, by F. Amato, M. Fonisto, M. Giacalone, and C. Sansone, 2023, Information, 14(6), Article 307. https://doi.org/10.3390/info14060307

For explicit skill comparison, JobMatch combine manually enter profile skills with confirm document-derived skills. Duplicate entries are removed through case-insensitive comparison. Only skills with a confirmed review status are included in this process, ensuring that pending or rejected document-derive suggestions are exclude from the confirmed-skill list.

The Python service encode candidate and vacancy text using the model.encode() method. The resulting embeddings are compare using cosine similarity. The raw similarity value is multiplied by 100 and restrict the range of 0 to 100 to obtain the semantic score. The final matching score is calculated using the following formula:

$$\text{Final Matching Score} = (0.80 \times \text{Semantic Score}) + (0.20 \times \text{Skill Coverage Score})$$

The missing required skills are treated as unmatched, while skill terms are normalised and compared using the implemented skill vocabulary and mappings, including documented equivalents such as "Entity Relationship Diagram" and "ERD". This hybrid approach integrates contextual text similarity and direct comparison of candidate skills against vacancy requirements.

The final score is then use to support job recommendations and recruiter-side candidate ranking. It represents a computational matching score rather than a calibrated probability of employment success. The implementation uses Sentence Transformers 6.0.1, Transformers 5.17.0, PyTorch 2.14.0, Python 3.14.6, and FastAPI 0.141.1 in the inspected development environment. The model's effective maximum input sequence length, text construction and preprocessing settings must be confirmed from the runtime configuration before being reported. Inputs exceeding the model's verified sequence-length limit should be described according to the actual truncation behaviour configured in the implementation.

**Recruitment workflows and data relationships**

Job seekers maintain profiles, search vacancies, inspect Best Matches, apply for positions, and monitor application status. Recruiters maintain company information, publish jobs, inspect applicant details and matching information, and shortlist candidates. MySQL links documents to users, OCR results and skill suggestions to documents, jobs to recruiters, and applications to the relevant candidate and vacancy. Persisted match_scores connect semantic outputs to recommendation and ranking screens.

This architecture demonstrates how one candidate representation can support discovery for a job seeker and screening for a recruiter. Nevertheless, the quality of either view depends on information completeness, the wording of vacancies, and the reliability of extraction and semantic processing. Final selection remains a human responsibility.

## RESULTS

**Functional and integration outcomes**

The main functional test table containing 14 scenarios all came back Pass, including registration, login, profile management, job creation, search, upload, OCR processing, skill assessment, recommendations, applications, tracking, applicant matching, and shortlisting. The main integration test table documented 12 scenarios and also came back Pass, covering the client-to-backend request route, data persistence, OCR skill processing, semantic service interaction, recommendations, applicant visibility, ranking, and shortlisting. These numbers pertain to those particular tables instead of all test entries found elsewhere in the report.

| Test category | Test details | Observed results |
|---|---|---|
| AI matching API | 10 candidate–job matching test cases | All 10 API requests returned HTTP 200 |

| | | |
|---|---|---|
| Ranking evaluation | Mean Precision@K and nDCG@K | 0.750 (75%) and 1.000 (100%), respectively |
| Skill extraction | Sample resume skill identification | 6 unique skills suggested; all remained pending review |
| Skill extraction unit tests | Skill identification, mapping, duplicate prevention, term distinction and empty input handling | All assertions passed in the displayed test run |
| OCR integration | PDF detection and page conversion | Completed successfully |
| OCR preprocessing | Enhanced-image and threshold-image comparison | Both returned confidence scores of 92 |
| OCR selection | Selection between the two OCR results | Enhanced image selected |
| Text extraction | Extraction of text from the processed resume PDF | Completed |
| Skill extraction integration | Identification of skills from extracted PDF text | 18 suggested skills; all remained pending review |

Five OCR scenarios, which are resume extraction, certificate extraction, supporting-document extraction, skill identification, and skill assessment, all came back as Pass. Five semantic scenarios are indicate as Successful, encompassing web development, Java and Android, database competencies, unrelated domain, and candidate details enhanced by verified document skills. The report offers qualitative anticipated behaviour instead of numerical retrieval metrics for these scenarios. No elapsed-time metrics or load-testing outcomes demonstrate throughput or production scalability.

The specific test scenario results indicated success and offer proof of the initial prototype's functional and integration viability. However, the tests fail to demonstrate complete system coverage, numerical OCR precision or unbiased recommendation and candidate-ranking efficiency. The OCR scenarios were not assessed with character or word error rates, and the semantic matching lacked an independently labelled candidate-to-job dataset for computing ranking metrics. Therefore, the findings should be viewed as an indication of the examined workflow behaviour rather than a complete validation of the algorithm's performance

**Job seeker feedback**

All seven job-seeker item means exceed 4.00 out of 5.00. The AI match scores helped respondents understand their suitability for a job, receiving the highest mean of 4.64, followed by application status tracking at 4.50, while registration and profile setup received 4.29. The results suggest positive perceptions among the 14 participating job seekers, with variation between workflow elements. Recommendation relevance in this role-specific table is 4.43; this is a different questionnaire item from the general matching-relevance score reported below.

| **Job seeker item** | **Mean out of 5** |
|---|---|
| Registration and profile setup | 4.29 |
| Job searching and browsing | 4.36 |
| Application status tracking | 4.50 |

| | |
|---|---|
| Match scores helped explain suitability | 4.64 |
| Recommendations reflected profile or resume | 4.43 |
| Interface and navigation clarity | 4.43 |
| Usefulness for job discovery | 4.43 |

**Recruiter and general feedback**

The two recruiter respondents assigned 5.00 out of 5.00 to each of the seven recruiter items. This covers account access, company profile management, job creation, applicant information, matching, ranking, and shortlisting. Although these responses indicate positive feedback from the participating recruiters, the uniformly maximum ratings may reflect a ceiling effect, limiting the ability to identify differences in their perceptions. These uniformly positive responses describe the two participants and cannot establish acceptance among recruiters generally.

| **General evaluation item** | **Mean out of 5** |
|---|---|
| Satisfactory system response | 4.31 |
| Features worked as expected | 4.44 |
| Clarity of displayed information | 4.44 |
| Ease of learning | 4.44 |
| Comfort using the system again | 4.38 |
| Overall satisfaction | 4.56 |

The report also gives 4.44 for perceived match-score accuracy and 4.50 for perceived matching or recommendation relevance. These are participant judgements. The reported overall satisfaction mean of 4.56 is an individual questionnaire item, not an average calculated from every item in this article. Because item-level response counts are not consistently supplied, the displayed means are reproduced as reported rather than recomputed.

**Preliminary OCR, skill extraction and semantic matching evaluation**

The semantic matching API test executed 10 test cases successfully and generated ranking metrics in output file jobmatch_ai_test_results.csv. The mean Precision@K (P@K) was 0.750 (75%), while the mean nDCG@K (Normalised Discounted Cumulative Gain) was 1.000 (100%). Moreover, the API returned matching scores ranging from 18.19 to 95.08 across the 10 test cases. The evaluation covered four job listings (J01–J04), with three candidates each for J01, J02 and J03, and only one candidate for J04.

Each candidate was assigned an illustrative relevance label from 0 to 3: 3 represented highly relevant, 2 represented relevant, 1 represented slightly relevant, and 0 represented not relevant. The dataset contained three candidates labelled 3, three labelled 2, no candidates labelled 1, and four labelled 0.

For P@K, candidates with relevance labels of 2 or higher were considered relevant. K was set to the smaller of five or the number of candidates available for each job. The per-job results are presented in the table below.

| Job listing | Ranked candidates (relevance labels) | K | P@K | nDCG@K |
|---|---|---|---|---|
| J01 | C01 (3), C03 (2), C02 (0) | 3 | 0.667 | 1.000 |
| J02 | C04 (3), C06 (2), C05 (0) | 3 | 0.667 | 1.000 |
| J03 | C07 (3), C09 (2), C08 (0) | 3 | 0.667 | 1.000 |
| J04 | C10 (3) | 1 | 1.000 | 1.000 |
| **Mean** | **Four job listings** | — | **0.750** | **1.000** |

The mean P@K indicates that, on average, 75% of the candidates in the evaluated top-ranked results met the test dataset's relevance threshold. Meanwhile, the mean nDCG@K of 1.000 indicates that the generated rankings followed the ideal ordering implied by the supplied relevance labels. These results can coexist because P@K measures the proportion of relevant candidates in the ranked results, whereas nDCG@K evaluates the ordering of candidates according to their graded relevance. However, J04 contained only one candidate, providing limited evidence of ranking ability. Furthermore, the relevance labels were illustrative and were not independently validated by human evaluators. Therefore, these metrics indicate consistency between the API rankings and the supplied test labels rather than proof of real-world recruitment effectiveness.

The skill extraction unit tests were conducted using the testSkillExtraction.js script, which contained four test groups. The first test used a sample resume to verify the identification of six expected skills: Python, MySQL, SQL, Database Normalization, ERD and Microsoft Excel. It also checked for duplicate skill names and confirmed that all extracted suggestions retained a pending-review status. The second test verified that the system could distinguish JavaScript from Java and GitHub from Git, preventing incorrect partial-word matches. The third test checked the handling of empty OCR text by confirming that zero skills were returned. Finally, the fourth test verified duplicate prevention when the same skill appeared multiple times in different forms. All assertions in the script passed in the reported test run, demonstrating that the tested functions behaved as expected for the supplied inputs. However, these results establish functional correctness for the selected test cases rather than overall skill extraction accuracy. Further evaluation using manually annotated resume data would be required to calculate extraction precision, recall and F1-score.

The OCR and skill-extraction integration test a PDF file, detecting the document, completed page conversion and text extraction, and returned 18 suggested skills. All 18 suggested skills remained pending review, consistent with the workflow in which users must review suggestions before they are used for matching. Moreover, enhanced-image OCR and threshold-image OCR both reported a confidence score of 92. The OCR confidence score of 92 is an engine-reported confidence value and should not be interpreted as 92% objective OCR accuracy; evaluating objective accuracy requires comparing extracted text with a manually checked reference transcription by calculating character error rate or word error rate. Similarly, the semantic-ranking metrics are limited because the test dataset uses illustrative relevance labels, and Job J04 has only one candidate. Independent human relevance judgements and a larger candidate-to-vacancy test set are required before these metrics can support stronger research claims.

## DISCUSSION

The findings indicate that JobMatch integrates document processing and semantic comparison into an operational mobile recruitment prototype. The functional, integration and semantic matching tests demonstrate the implementation of key workflows, from document upload and skill extraction to user confirmation and candidate-to-job matching. Furthermore, questionnaire responses suggest that participating users perceived the platform as useful for job discovery and recruitment activities. The skill-confirmation feature allows users to review and approve extracted skills before they are used for matching, helping to reduce the inclusion of incorrect or irrelevant profile information. The questionnaire results also highlight areas for improvement, as

job seekers rated registration and profile setup at 4.29, while the helpfulness of AI match scores received a higher mean of 4.64; the general system-response rating was 4.31. These findings can guide future improvements to usability and system responsiveness, although further observation and performance measurements are needed to identify specific causes. Nevertheless, the semantic matching evaluation does not establish superiority over conventional keyword-based approaches. Future work should compare keyword-based matching, TF-IDF, semantic-only, skill-only and hybrid scoring using the same independently labelled candidate-to-job dataset. Evaluating profiles before and after users confirm extracted skills could also help determine the contribution of the review process to matching performance. Overall, JobMatch should be considered a recruitment decision-support prototype rather than an automated hiring system, as candidate suitability requires human judgement and consideration of factors beyond resume and job-description content.

The skill-confirmation stage offers a useful design contribution since the automatically extracted text can contain mistakes, and not every phrase in a document represents an endorsed candidate skill, so persisting suggestions separately from confirmed skills makes that distinction explicit. Future evaluation could also compare the matching behaviour before and after user review to quantify whether the checkpoint reduces irrelevant or erroneous profile content.

The role-specific result identifies practical areas for refinement since registration and profile setup scored lower than match-score helpfulness among job seekers, while system response scored 4.31 in the general evaluation. These means can guide follow-up interviews and task observation, but they do not identify specific causes. It would be premature to attribute the response rating to network latency or a particular service without measurement.

The semantic scenarios do not demonstrate superiority over keyword search. A credible comparison would require the same candidate-to–job dataset, consistent preprocessing, and explicit baselines. Likewise, high perceived relevance does not establish that the system ranks the most suitable applicants first. Human assessment, actual job requirements, and factors absent from document text may all affect suitability. The prototype should therefore be interpreted as decision support.

**Limitations and Future Work**

The study has several limitations that should be considered when interpreting the findings. The small pilot sample, particularly the two recruiter respondents, limits the generalizability of the questionnaire results. The absence of raw response data also prevents further assessment of response variability and potential ceiling effects. In addition, the evaluation did not include controlled comparisons with conventional matching methods, long-term user adoption or actual hiring outcomes. Therefore, the findings cannot establish the system's overall recruitment effectiveness, fairness or security assurance, or usability across users with different levels of digital access. Future work should involve a larger and more diverse participant sample, independently labelled candidate–job datasets, controlled matching comparisons and longer-term evaluation. Further investigation should also document the system's data access and retention controls and establish the applicable consent and ethics-review procedures.

Further OCR evaluation to compare extracted text with manually checked transcriptions, reporting character error rate and word error rate across document types, scan quality, and layouts. Moreover, the skill extraction should be evaluate separately through label skill annotations and precision, recall and F1 measures. This would distinguish recognition errors from errors in selecting skills from correctly recognized text.

Future enhancement for semantic evaluation should use independently labelled candidate-to-vacancy mappings pairs, a documented relevance rubric and a TF-IDF baseline. Additionally, ranking measures such as Precision at K and normalized discounted cumulative gain would assess ordering quality. A removal comparison of profile-only text with profile plus confirm document skills would isolate the effect of document enrichment. All of these are proposed evaluations, not completed experiments.

Reproducibility could be improved by documenting the exact embedding model identifier, dependency versions, input construction rules, text truncation strategy, score transformation method, and test dataset composition. Deployment work should measure service latency and failure handling, review document access controls and retention, and evaluate ranking behaviour across relevant applicant groups. Participant consent and any institutional ethics requirements must be documented for subsequent research; this article does not assume that formal approval was obtained.

## CONCLUSION

JobMatch demonstrates the integration of OCR, user-confirmed skill extraction, semantic text comparison and recruitment management within an Android platform. The FYP evaluation recorded successful execution of the listed functional and integration test scenarios, together with positive pilot feedback from 14 job seekers and 2 recruiters. The overall satisfaction mean was 4.56 out of 5.00, indicating preliminary perceived usefulness among the participants. These findings support the feasibility of developing an integrated mobile recruitment prototype. However, the study does not establish validated extraction accuracy, recommendation effectiveness, fairness, scalability or actual employment impact. Further evaluation using larger datasets, independent assessments and controlled comparisons is necessary to determine the system's effectiveness and reliability in real-world recruitment settings.

## REFERENCES

1. Amato, F., Fonisto, M., Giacalone, M., & Sansone, C. (2023). An intelligent conversational agent for the legal domain. Information, 14(6), Article 307. https://doi.org/10.3390/info14060307
2. Smith, R. (2007). An overview of the Tesseract OCR engine. In Proceedings of the Ninth International Conference on Document Analysis and Recognition (Vol. 2, pp. 629–633). IEEE. https://doi.org/10.1109/ICDAR.2007.4376991
3. Fabris, A., Baranowska, N., Dennis, M. J., Graus, D., Hacker, P., Saldivar, J., Zuiderveen Borgesius, F., & Biega, A. J. (2025). Fairness and bias in algorithmic hiring: A multidisciplinary survey. ACM Transactions on Intelligent Systems and Technology, 16(1), Article 16. https://doi.org/10.1145/3696457
4. Gugnani, A., & Misra, H. (2020). Implicit skills extraction using document embedding and its use in job recommendation. Proceedings of the AAAI Conference on Artificial Intelligence, 34(8), 13286–13293. https://doi.org/10.1609/aaai.v34i08.7038
5. Muhammad Ashraf, F. N. H. (2026). *JobMatch: An intelligent platform for smart recruitment and job discovery* [Unpublished undergraduate final year project report]. Quest International University.
6. Kelan, E. K. (2024). Algorithmic inclusion: Shaping the predictive algorithms of artificial intelligence in hiring. Human Resource Management Journal, 34(3), 694–707. https://doi.org/10.1111/1748-8583.12511
7. Job recommender systems: A systematic literature review, applications, open issues, and challenges. (2025). Journal of Big Data. https://link.springer.com/article/10.1186/s40537-025-01173-y
8. Köchling, A., & Wehner, M. C. (2020). Discriminated by an algorithm: A systematic review of discrimination and fairness by algorithmic decision-making in the context of HR recruitment and HR development. Business Research, 13, 795–848. https://doi.org/10.1007/s40685-020-00134-w
9. Ochmann, J., Michels, L., Tiefenbeck, V., Maier, C., & Laumer, S. (2024). Perceived algorithmic fairness: An empirical study of transparency and anthropomorphism in algorithmic recruiting. Information Systems Journal, 34(2), 384–414. https://doi.org/10.1111/isj.12482
10. Raghavan, M., Barocas, S., Kleinberg, J., & Levy, K. (2020). Mitigating bias in algorithmic hiring: Evaluating claims and practices. In Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency (pp. 469–481). Association for Computing Machinery. https://doi.org/10.1145/3351095.3372828
11. Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-

IJCNLP) (pp. 3982–3992). Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1410

12. Representing jobs and job seekers in AI-based recommender systems: A literature review. (2026). Engineering Applications of Artificial Intelligence. https://doi.org/10.1016/j.engappai.2026.114450
13. Vanetik, N., & Kogan, G. (2023). Job vacancy ranking with sentence embeddings, keywords, and named entities. Information, 14(8), Article 468. https://doi.org/10.3390/info14080468