International Scientific Journal of Engineering and Management (ISJEM)
Volume: 05 Issue: 04 | April – 2026
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata
ISSN: 2583-6129
DOI:10.55041/ISJEM06414

Recruitment Through AI Tech Agent

Dr. Y. H. Prasanna Raju, P. Meher Srija, L. Meenakshi, S. Ganesh, T. Vinay
prasannaraju@mvgrce.edu.in | potnurumehersrija@gmail.com | lalammeenakshi5@gmail.com | ganasiddabattula@gmail.com | tavvavinay764@gmail.com
Department of Information Technology, MVGR College of Engineering (A), Vizianagaram, Andhra Pradesh

Abstract — Recruitment plays a vital role in every organization, but traditional hiring methods often come with several limitations. Manual screening of resumes is time-consuming, interview questions may differ for each candidate, and evaluations can sometimes be subjective. These challenges may slow down the hiring process, increase overall costs, and lead to inconsistent or unfair assessment of candidates.

This paper introduces Recruitment through AI Tech Agent, a smart web-based platform that automates the entire hiring process. Organizations can post job roles along with required skills, and the system automatically selects customized interview questions using structured domain-specific datasets, Candidates can participate in fully proctored online interviews through text or voice. Their responses are evaluated using a hybrid scoring method, which includes direct matching for multiple-choice questions, test-case execution for coding problems, and semantic evaluation using LLMs (Ollama) for descriptive answers.

The system backend is developed using Python and Flask, while MongoDB is used for data storage. The results show that the platform provides accurate and consistent candidate evaluation with reduced bias. It also helps in significantly decreasing the workload of recruiters and offers a practical, scalable solution for modern recruitment needs.

Keywords — *Artificial Intelligence, Recruitment Automation, NLP, Large Language Models, Flask, MongoDB, Online Assessment, Proctoring.*

I. INTRODUCTION

Recruitment is one of the most time-consuming processes in any organization. Traditional hiring methods, such as manual resume screening, conducting live interviews, and subjective evaluation, are often slow and costly. These approaches can also introduce unconscious bias, making it difficult to select the most suitable candidates in a fair and consistent manner.

The fast development of Artificial Intelligence (AI) and Natural Language Processing (NLP) has opened new possibilities for improving the recruitment process. Large Language Models (LLMs), which are trained on large amounts of text data, are capable of understanding language, analyzing meaning, and evaluating responses based on context. These abilities make them suitable for automating interview assessment and improving candidate evaluation.

This paper presents **Recruitment through AI Tech Agent**, a complete AI-based platform that allows organizations to define job requirements. Based on these requirements, the system automatically generates a customized set of interview questions for each candidate session. Candidates participate in a fully automated interview that is monitored through a webcam, and their responses are evaluated using a multi-modal scoring engine powered by NLP and LLM technologies. The platform is developed using Python, Flask, MongoDB, and Ollama, and it supports multiple question types including MCQs, conceptual questions, and coding challenges.

LITERATURE SURVEY

Applications of AI and NLP in recruitment automation have gained significant attention in recent years. The following survey discusses the most relevant existing works and highlights the limitations that are addressed by the proposed system.

[1] **Rajput et al. (2024):** This work presents an AI chatbot designed to conduct both technical and HR interviews using machine learning classifiers. The system improves accessibility and automates basic interview interactions. However, it does not support coding-based assessments and lacks proper benchmark evaluation.

[2] **Jain et al. (2024):** The authors proposed an NLP-based chatbot that evaluates answers using keyword and pattern matching techniques. While the system can handle structured responses, it struggles to accurately assess open-ended conceptual answers due to its shallow evaluation mechanism.

[3] **Yadav et al. (2025):** This study introduces a virtual interview system that uses speech-to-text technology along with basic semantic feedback. Although it supports voice-based interaction, the system does not include coding evaluation or any proctoring mechanism.

[4] **Durga et al. (2023):** This paper presents a machine learning classification approach combined with a rule-based chatbot for delivering interview questions. The system lacks real-time adaptability and does not support coding-based answer submissions.

[5] **Bhavana et al. (2023):** The authors developed a mock interview evaluation system that accepts both text and voice input and performs feature extraction for

assessment. However, the system is tested on limited data and does not include interview proctoring.

[6] **Jurafsky & Martin (2023):** This work provides foundational concepts in natural language processing that support semantic evaluation techniques. It emphasizes attention-based context understanding, which is useful for comparing candidate responses meaningfully.

[7] **Vaswani et al. (2017):** This paper introduces the Transformer architecture, which forms the basis of modern large language models. The self-attention mechanism enables better understanding and comparison of descriptive answers.

[8] **Goodfellow et al. (2016):** This work explains the core principles of deep learning that support language model architectures. These concepts are widely used in building AI systems for automated response evaluation.

The reviewed literature highlights four common limitations. First, many systems support only a single type of question and do not include coding-based assessments. Second, semantic evaluation is either missing or too basic to properly assess descriptive answers. Third, most existing approaches do not provide integrated real-time proctoring. Finally, there is no complete end-to-end platform that handles the entire recruitment process. The proposed system addresses all of these limitations.

II. PROPOSED METHODOLOGY

The system automates the complete technical recruitment process through an integrated web-based platform. It is organized into several stages, including registration and authentication, job configuration, dataset-based question generation, proctored interview conduction, multi-modal response evaluation, weighted scoring, and final result reporting.

A. System Architecture

The system follows a three-tier architecture consisting of a web-based frontend, a Flask-based backend, and MongoDB for data storage. The Flask backend manages the overall functionality through different sub-modules, including the Dataset Manager, Question Generator, AI Answer Scoring Engine, and Face/Voice Monitor. LLM inference is performed locally using Ollama, which eliminates the need for external APIs. The overall system architecture is illustrated in Fig. 1.

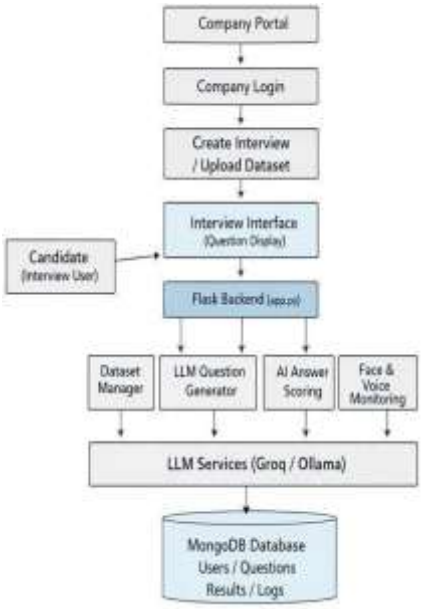

Fig. 1: System Architecture — Working Methodology Block Diagram

B. User Registration & Authentication

Both companies and candidates register on the platform and choose their respective roles. Authentication and role-based access control are handled using Flask-Login. Sensitive information, such as credentials, is securely managed through environment variables using python-dotenv.

C. Job Posting and Configuration

After logging in, companies can create job postings by providing details such as job title, description, required domains (for example, Python, SQL, Java), interview duration, question types, and the qualifying cut-off score. These settings control the subsequent stages of the recruitment workflow.

D. Dataset Structure & Question Generation

The question repository is organized into domain-specific CSV files that include the question text, reference answer, question type (MCQ, conceptual, or coding), and options for MCQs. When an interview session begins, the system filters and randomly selects questions based on the configured interview duration. Companies also have the option to upload their own custom datasets.

E. Proctored Interview Delivery

Questions are displayed one after another, and candidates can submit responses through text or voice. The webcam feed is continuously monitored during the interview. Anti-malpractice measures are applied, such as terminating the exam after more than three tab switches and closing the session if the candidate is absent from the camera for an extended period. The process flow is illustrated in Fig. 2.

International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Volume: 05 Issue: 04 | April – 2026 DOI:10.55041/ISJEM06414
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

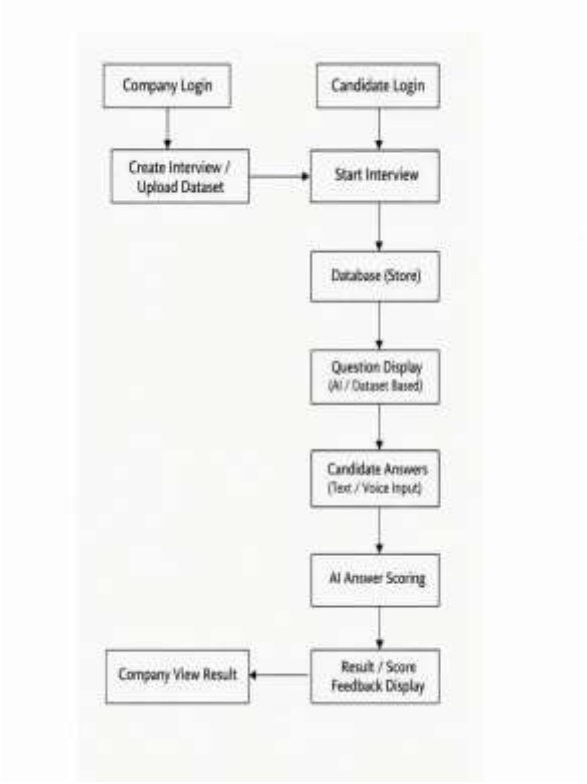

Fig. 2: Process Flowchart of the AI Recruitment System

F. Multi-Modal Response Evaluation

A hybrid scoring engine is used to evaluate different types of responses using suitable methods. Multiple-choice questions are assessed through direct answer matching. Coding responses are executed against predefined test cases, and marks are assigned proportionally based on correctness. Descriptive answers are evaluated using a local LLM through Ollama, which analyzes semantic similarity and awards marks based on conceptual correctness rather than exact wording.

A weighted scoring scheme is also applied. MCQ and conceptual questions are assigned a weight of 1, while coding questions are given a weight of 2 due to their higher complexity. An example of this scoring configuration is shown in Table I.

Table I: Example Weighted Scoring Configuration

Question Type	**Count**	**Weight**	**Max Scor e**
MCQ	3	1	3
Conceptual	2	1	2
Coding	2	2	4
Total			**9**

G. Qualification and Result Reporting

The final percentage is calculated by dividing the total obtained score by the maximum possible score and multiplying by 100. Candidates who achieve a score equal to or above the specified cut-off are marked as **HR Eligible**, while the remaining candidates are given a **Fail** status. The results are stored in MongoDB and displayed on both the candidate and recruiter dashboards.

III. RESULTS

The platform was tested with multiple candidate sessions in Python and SQL domains. The interview configuration included a 30-minute duration, a 70% qualifying cut-off, and a mix of MCQ, conceptual, and coding questions.

A. System Interface

The homepage (Fig. 3) provides separate entry options for companies and candidates, along with key features such as AI-based evaluation, webcam monitoring, and support for multiple domains. Role-based access control is implemented to ensure that users cannot access unauthorized pages.

Fig. 3: Website Homepage — AI Interview Platform

B. Company Dashboard

The company dashboard (Fig. 4) displays all posted job roles along with their domains, interview durations, and number of applicants. During testing, a Software Engineer position configured with Python and SQL, a 30-minute duration, and a 70% cut-off showed 4 candidate submissions, with 1 candidate marked as HR eligible and an average score of 41.2.

Fig. 4: Company Dashboard

ISSN: 2583-6129
DOI:10.55041/ISJEM06414

C. Student Dashboard and Submission

The student dashboard (Fig. 5) displays available job opportunities along with domain and duration details, as well as the candidate's progress. After completing the interview, candidates can immediately view their score, percentage, and HR eligibility status. The submissions panel (Fig. 6) presents all candidate entries, including email, score, eligibility, and review status.

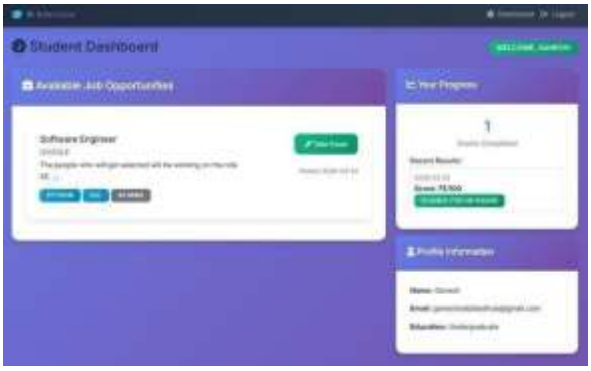

Fig. 5: Student Dashboard

Fig. 6: Student Submissions Panel

D. Evaluation Results

Table II presents the results of the Software Engineer test scenario. Four candidates participated in the interview, and the system accurately determined pass and fail outcomes. The scores ranged from 0 to 75 out of 100, indicating clear differences in candidate performance.

Table II: Candidate Evaluation Results (Software Engineer — Python Domain)

Candidate	Domain	Score	Cut-of f	Status
Ganesh	Python	75/100	70%	PASS (HR Eligible)
Tavva Vinay	Python	57/100	70%	FAIL
Meenakshi	Python	0/100	70%	FAIL
Srija	Python	33/100	70%	FAIL

The LLM-based semantic evaluation module successfully assigned partial marks for answers that were conceptually correct but worded differently from the reference answers, performing better than simple keyword matching approaches. The test-case execution module also applied proportional scoring accurately for coding responses. Additionally, the system maintained stable performance during concurrent sessions, with no data conflicts or noticeable performance issues.

IV. LIMITATIONS

The following limitations were identified for future improvement and deployment considerations. The question datasets are manually prepared, which limits their coverage and scalability. LLM-based evaluation may experience increased response time when many candidates are assessed simultaneously. The coding module currently uses Python exec, which can introduce security risks if proper sandboxing is not implemented. Webcam proctoring can detect candidate absence but does not verify identity through facial recognition. The system currently supports only text-based technical roles, and the performance of the LLM may decrease for highly specialized domains that are not well represented in its training data.

V. FUTURE SCOPE

Several enhancements are planned for future development. These include integrating BERT or sentence-transformer embeddings to improve semantic evaluation. A Docker-based sandboxed environment is proposed for secure coding execution, with support for languages such as Java, C++, and JavaScript. The system can also be extended to include video-based assessment with facial expression analysis and speech sentiment detection.

Further improvements involve expanding domain coverage and difficulty levels in the dataset, deploying the platform on cloud services like AWS or Azure for better scalability, and integrating with Applicant Tracking Systems (ATS) along with automated interview scheduling. Additional features include recruiter analytics dashboards with AI-generated hiring recommendations, as well as enhanced proctoring using facial recognition and gaze tracking.

CONCLUSION

This paper presented **Recruitment through AI Tech Agent**, a complete AI-powered platform that automates and standardizes the technical hiring process. The system uses dynamic question generation, real-time webcam

ISSN: 2583-6129
DOI:10.55041/ISJEM06414

proctoring, and a multi-modal hybrid scoring engine. Multiple-choice questions are evaluated through direct matching, coding responses are assessed using test-case execution, and descriptive answers are analyzed using LLM-based semantic evaluation.

Experimental results show accurate candidate evaluation across all question types, fair weighted scoring, and correct qualification decisions. All system modules were successfully validated during concurrent sessions. The platform is built using Python, Flask, MongoDB, and Ollama, and can be deployed on standard hardware. The AI Tech Agent demonstrates a practical approach to intelligent recruitment and highlights the real-world usefulness of NLP and LLM technologies in modern organizational workflows.

REFERENCES

[1] Rajput et al., "AI Interviewer Chatbot for Technical and HR Interview Automation," International Journal of Innovative Research in Computer Science and Technology, 2024.

[2] K. Jain et al., "Design and Implementation of an Automated Interview System using NLP Chatbots," IEEE International Conference on AI and Machine Learning, 2024.

[3] Yadav et al., "Virtual AI-Based Interview System with Speech Integration and Automated Feedback," Journal of Emerging Technologies in Computing, 2025.

[4] Durga et al., "Intelligent Interview Assistant using Machine Learning and Chatbot Frameworks," International Journal of Computer Applications, vol. 185, no. 12, 2023.

[5] Bhavana et al., "AI-Based Mock Interview Evaluator with Feature Extraction Techniques," IEEE Access, 2023.

[6] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, 3rd ed., Pearson Education, 2023.

[7] A. Vaswani et al., "Attention Is All You Need," Advances in Neural Information Processing Systems (NeurIPS), 2017.

[8] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[9] Flask Documentation, Pallets Projects. Available: https://flask.palletsprojects.com/

[10] MongoDB Documentation, MongoDB Inc. Available: https://www.mongodb.com/docs/

[11] Ollama Documentation. Available: https://ollama.com/

[12] Python Software Foundation, Python 3.10 LanguageReferenceAvailable: https://docs.python.org/3/

[13] Pandas Development Team, Pandas Documentation.Available: https://pandas.pydata.org/docs