AI-Enabled Automated Interview Evaluation System

Vaibhavi Kabade[1], Gayatri Patil[2], Shivani Godse[3], Aayusha Jain[4], Prof. Mayur Kumbharde[5]

[1,2,3,4]Student, SNJB's Late Sau Kantabai Bhavarlalji Jain College of Engineering, Nashik
[5]Faculty of SNJB's Late Sau Kantabai Bhavarlalji Jain College of Engineering, Nashik

Abstract

In today's competitive job market, evaluating candidates efficiently and objectively is crucial for recruiters. This paper presents the design and implementation of an AI-enabled Interview Evaluation System that automates candidate assessment through a combination of multiple intelligent modules. It includes MCQ-based technical evaluations, semantic analysis of verbal responses using NLP models, and real-time facial verification via computer vision. The system is built using Flask for backend processing and SQLite as the database, offering a lightweight yet powerful solution for small-to-mid-scale deployment. Verbal answers are evaluated using transformer-based semantic similarity models from the Sentence Transformer library. Webcam-based face validation is performed using OpenCV and Haar Cascade classifiers to detect malicious activity such as impersonation or multiple faces. The system supports both typed and spoken answers, includes PDF report generation, and provides an intuitive admin dashboard. Experimental results demonstrate the model's efficiency and accuracy in scoring responses and maintaining interview integrity.

Keywords: Interview Bot, Natural Language Processing, Verbal Evaluation, MCQ Assessment, Face Detection, Flask, Sentence Transformer, Haar Cascade.

I. INTRODUCTION

The process of recruiting and selecting the right candidates is fundamental to building a competent and productive workforce. Human Resource (HR) departments around the globe invest substantial time, effort, and financial resources to screen, evaluate, and hire talent that aligns with organizational goals. However, the conventional hiring process, which typically involves resume shortlisting, manual interviews, and panel-based evaluations, suffers from a range of inefficiencies and biases that impact decision-making quality.

Traditional interview techniques are inherently subjective and often vary based on the interviewer's perspective, mood, or experience. This variability introduces inconsistency in candidate evaluation, where two equally qualified individuals may receive different outcomes based solely on who interviews them. Furthermore, interviewer fatigue, especially during mass hiring drives, leads to oversight of potential talent. The sheer volume of applications received during open recruitment sessions in universities or large corporations makes it impractical to provide every candidate with equal attention and unbiased evaluation.

Moreover, logistical challenges such as scheduling interviews, coordinating interviewer availability, and managing records further complicate the recruitment process. The need for an objective, scalable, and technology-driven system has become more evident in the post-pandemic era, where remote hiring and digital interactions have become the norm.

To address these challenges, this paper introduces an AI-powered Automated Interview Evaluation System that intelligently combines various modern technologies to simulate and automate the initial stages of candidate assessment. The system encompasses:

Candidate Registration and Resume Upload: A web-based interface for collecting essential applicant data.

Technical Knowledge Assessment via MCQs: A domain-specific quiz mechanism that tests core knowledge using a bank of preloaded questions.

Verbal Communication Assessment: An intelligent verbal interview simulation using Natural Language Processing (NLP) to score answers semantically.

Face Detection and Integrity Checks: Integration of computer vision to detect presence, absence, or multiplicity of faces during the interview, ensuring identity validation and discouraging impersonation.

The backend of the system is built using Flask, a lightweight Python web framework, coupled with SQLite as a local database solution for efficient data management. The system leverages transformer-based language models from the SentenceTransformer family to analyze and semantically compare candidate answers against ideal model responses. This eliminates the need for manual evaluation while preserving fairness, accuracy, and interpretability.

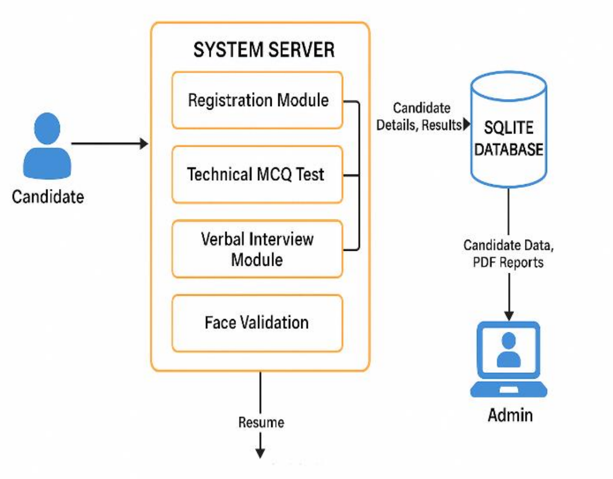
On the visual surveillance front, the platform incorporates OpenCV's Haar Cascade classifier to continuously monitor the candidate's webcam feed. By analyzing facial data in real time, the system raises alerts when multiple faces or no faces are detected, thereby discouraging cheating, collaboration, or substitution during remote interviews.

By unifying NLP, computer vision, and a modular web interface, the system creates a semi-automated interview platform that can process dozens or hundreds of applicants with minimal human supervision. HR administrators can log in through a dedicated admin portal to review candidate performance, download performance reports in PDF format, and manage interview data securely and efficiently.

Ultimately, the goal of this system is to enhance transparency, scalability, and objectivity in recruitment processes. By automating repetitive tasks and relying on AI-based scoring mechanisms, organizations can reduce bias, optimize recruitment costs, and accelerate decision-making—all while offering candidates a structured, modern interview experience. This research aims to bridge the traditional human-centered interviewing system with an AI-driven future, promoting merit-based hiring at scale.

II. SYSTEM OVERVIEW

The AI-Enabled Automated Interview Evaluation System is designed as a modular, web-based application capable of handling the end-to-end screening process for candidates in a semi-automated manner. The system architecture is divided into four core components: Candidate Registration, Technical MCQ Evaluation, Verbal Interview Analysis, and Face Detection for Identity Validation. Together, these modules ensure a comprehensive and unbiased assessment of candidates through a seamless digital workflow. Additionally, an Admin Panel is provided for centralized monitoring and report generation.



A. Registration Module

The process begins with the Registration Module, which presents a simple and user-friendly form for applicants to submit their personal details, including full name, email ID, and educational institution. Candidates are also required to upload their resume in PDF format, which is stored securely on the server and can later be reviewed by administrators. This module ensures a standardized and digital record of all participating candidates, eliminating the need for manual data entry.

To enhance usability, the registration page provides a responsive interface that validates input fields and prevents form submission unless all required data is entered. Once registered, the candidate is uniquely identified using their email address and redirected to the role selection screen to initiate the assessment process.

B. Technical MCQ Test Module

Following registration, the candidate selects a specific job role (e.g., Python Developer, Java Developer, Full Stack Web Developer, etc.) which determines the set of technical questions they receive. The MCQ (Multiple Choice Questions) Module loads these role-specific questions from a CSV dataset using Flask server-side logic. Each question comes with four randomized options, and only one correct answer.

The test interface allows candidates to navigate between questions using “Next” and “Previous” buttons, and a final Submit button is displayed at the end of the test. The system computes the total score by matching user responses with the correct answers and stores the score in the MCQResult database table. Candidates scoring above a predefined threshold (e.g., 6 out of 10) qualify for the next phase: the verbal interview.

C. Verbal Interview Module

In this module, candidates are presented with a set of open-ended technical and behavioral questions. They are allowed to respond either by typing or speaking via integrated voice recognition using the Web Speech API. Each response is semantically evaluated using pre-trained transformer models from the SentenceTransformer family (paraphrase-MiniLM-L6-v2).

The evaluation is done by comparing the candidate’s answer with an ideal or model answer using cosine similarity of vector embeddings. A similarity score is generated for each response, which is then scaled to a 10-point mark and stored in the VerbalAnswer table. This enables detailed insight into the candidate’s comprehension, communication, and reasoning capabilities.

D. Face Validation Module

To ensure the integrity of the evaluation, the system incorporates a real-time face validation mechanism using the candidate’s webcam. This module captures frames every few seconds and applies Haar Cascade classifiers using the OpenCV library to detect faces.

The system identifies three scenarios:

1. Single Face Detected – Valid condition, interview proceeds.
2. No Face Detected – Potential candidate absence; an alert is triggered.
3. Multiple Faces Detected – Possible impersonation or external interference; system warns and logs the event.

This live face monitoring discourages fraudulent behavior during the online interview and provides confidence in the fairness of the process.

E. Admin Panel and Report Generation

A secure admin login portal allows authorized personnel to:

View all registered candidates

Review individual scores and answers

Access uploaded resumes

Search candidates by ID

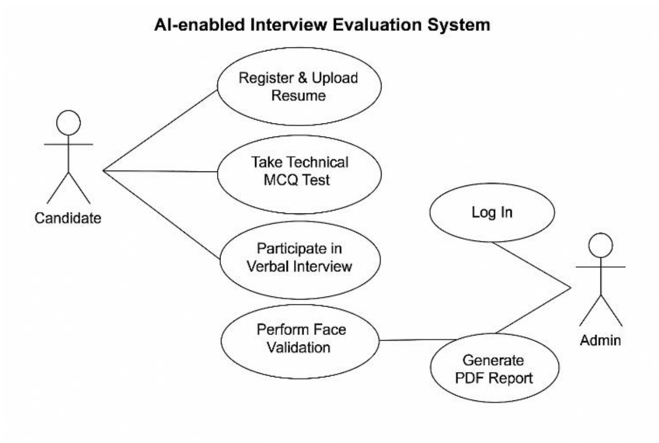
Generate PDF reports

Reports contain detailed performance metrics, including MCQ and verbal scores, semantic observations, and an overall interview outcome (e.g., *Recommended* or *Needs Improvement*). Two types of reports are generated:

Candidate Report – For applicant reference

Admin Report – Includes additional analytics such as accuracy rate and performance recommendations

All reports are generated dynamically using the FPDF library and made available for download or preview within the dashboard.



III. TECHNOLOGIES USED

The proposed AI-enabled Interview Evaluation System is a full-stack application that brings together multiple technologies from the fields of web development, artificial intelligence (AI), natural language processing (NLP), and computer vision. Each component plays a critical role in achieving automation, scalability, and precision in the interview process. The following technologies were used in the implementation:

A. Backend: Flask (Python)

The server-side logic of the system is developed using **Flask**, a lightweight and flexible Python web framework. Flask allows for rapid prototyping and provides built-in support for routing, form handling, template rendering (via Jinja2), and session management. The modular nature of Flask makes it suitable for integrating AI models, APIs, and database operations within a single application architecture. Additionally, Flask's compatibility with Python machine learning libraries enables seamless embedding of NLP and face detection functionalities.

B. Database: SQLite

For data persistence, **SQLite** is used as the backend relational database management system. It stores all essential data, including candidate information, MCQ responses, verbal answers, evaluation scores, and resume file paths. SQLite is lightweight, requires no separate server installation, and is highly reliable for standalone applications and small-scale deployments. Flask's SQLAlchemy ORM (Object Relational Mapper) is used to simplify database interactions in a Pythonic manner.

C. Frontend: HTML, CSS, JavaScript

The user interface (UI) for both candidates and administrators is developed using standard web technologies:

HTML5: Provides the structural framework for all forms and display elements.

CSS3: Used for styling and animations, enhancing usability with responsive designs.

JavaScript: Implements dynamic behavior, including speech recognition, webcam integration, and real-time validations. JavaScript also handles alert messages for face detection feedback and improves the user experience through smooth UI transitions.

D. NLP Model: SentenceTransformer

The core component responsible for verbal answer evaluation is a **pre-trained transformer-based NLP model** from the SentenceTransformer library. The model used, **paraphrase-MiniLM-L6-v2**, generates high-quality sentence embeddings that capture semantic meaning. Candidate answers are compared with reference answers using **cosine similarity**, enabling an objective scoring mechanism based on how closely the response matches the expected content. This approach reduces subjectivity and enhances fairness in the verbal interview process.

E. Face Detection: OpenCV's Haar Cascade Classifier

To maintain interview integrity, the system uses **OpenCV**, an open-source computer vision library, to perform face detection. Specifically, the **Haar Cascade classifier** is employed to identify faces in real-time using frames captured from the candidate's webcam. The system checks for the presence of:

No face (indicating absence or misuse),

Multiple faces (indicating potential cheating),

Single valid face (allowing continuation).

These checks help maintain the credibility of remote interviews without requiring a human proctor.

F. PDF Report Generation: FPDF

The platform automatically generates **PDF performance reports** for both candidates and administrators using the **FPDF** Python library. These reports include:

Candidate details

MCQ and verbal scores

Semantic observations

Final recommendations

FPDF provides flexibility in formatting and supports multilingual text, making it ideal for generating formal, printable interview reports.

G. Voice Input: Web Speech API (SpeechRecognition)

The system supports **voice-based responses** in the verbal interview module. This is achieved through the **Web Speech API**, a browser-based JavaScript interface that captures spoken input and transcribes it to text in real time. The feature provides accessibility for users who prefer speaking over typing and enhances realism in simulated interviews. Voice input can be manually started and stopped using on-screen buttons, ensuring user control over the recording process.

IV. IMPLEMENTATION DETAILS

The implementation of the AI-enabled Interview Evaluation System involves the seamless integration of frontend interfaces, backend logic, AI-based scoring mechanisms, and data validation tools. The system is designed to provide a structured, step-by-step candidate evaluation workflow from registration to final report generation. The following sub-sections describe the implementation components in detail:

A. Registration & Resume Upload

The entry point of the system is the candidate registration module. The **frontend form** collects essential details including **name**, **email**, and **college** information. A secure file input is provided to upload a **resume in PDF format**, which is stored in a dedicated server directory (/static/resumes/) with a unique filename based on the candidate's email ID.

The **backend logic**, implemented in Flask, handles form validation, file storage, and database entry creation. A new entry is created in the Candidate table of the SQLite database, ensuring each user is uniquely identified and traceable throughout the evaluation process. This data is later used for role-based redirection, performance mapping, and reporting.

B. MCQ Test Evaluation

After registration, the candidate selects a **specific job role** from a predefined list (e.g., Python Developer, Full Stack Web Developer). Based on the selected role, the system dynamically loads a set of **10 multiple-choice questions (MCQs)** from a role-tagged CSV file using Flask and Python's CSV module.

Upon submission:

The system compares **candidate-selected answers** with the correct options.

A **total score** out of 10 is computed.

Each attempt, including the candidate's ID, total score, and number of questions, is stored in the MCQResult table.

Candidates scoring ≥ **6/10** are allowed to proceed to the verbal interview phase.

A user-friendly interface built with HTML, CSS, and JavaScript allows smooth navigation between questions using"Next" and "Previous" buttons, enhancing the candidate experience.

C. Verbal Answer Evaluation

Qualified candidates proceed to a **verbal interview phase**, where they answer a set of **technical and behavioral questions**. These responses can be entered via:

Text typing

Voice transcription (enabled using Web Speech API)

For each question:

1. The system retrieves a **model answer** for comparison.
2. Both the candidate's answer and the reference answer are passed through the **SentenceTransformer model** to generate sentence embeddings.
3. **Cosine similarity** is calculated between the two embeddings to assess semantic closeness.
4. The result is converted to a **score out of 10**, which is stored in the VerbalAnswer table.
5. The process repeats until all questions are answered.

The final **verbal score** is the aggregate of all individual scores, up to a maximum of **30 points**.

D. Face Detection Integration

To ensure interview integrity, the system continuously monitors the candidate's **webcam feed** using the navigator.mediaDevices.getUserMedia() API combined with a hidden **canvas rendering pipeline**. Every 4 seconds, a snapshot is taken and:

Converted to base64

Sent to the /detect_faces Flask endpoint

Analyzed using **OpenCV's Haar Cascade classifier**

The system identifies the number of faces present:

Single Face Detected: The interview continues normally.

No Face Detected: A warning is shown indicating candidate absence.

Multiple Faces Detected: A cheating alert is triggered.

These alerts are handled in real-time using JavaScript alerts to prompt the candidate to correct their behavior.

E. Report Generation

Upon completing both MCQ and verbal phases, a comprehensive **performance report** is generated. There are two types of reports:

1. **Candidate Report** – Accessible on the final "Thank You" page, this report includes:

   Candidate's name, email, college

   MCQ and verbal scores

   Final outcome: Recommended or Needs Improvement

2. **Admin Report** – Generated via the admin dashboard with additional analytics:

   Accuracy rate in MCQs

   Semantic performance rate in verbal answers

   Observations: e.g., "Strong grasp of technical skills", "Struggles with complex problems"

   Recommendation summary

All reports are created dynamically using the **FPDF library** and saved as PDFs within a /static/reports/ directory. Admins can view these in-browser (via iframe) and also download them.

V. RESULT ANALYSIS

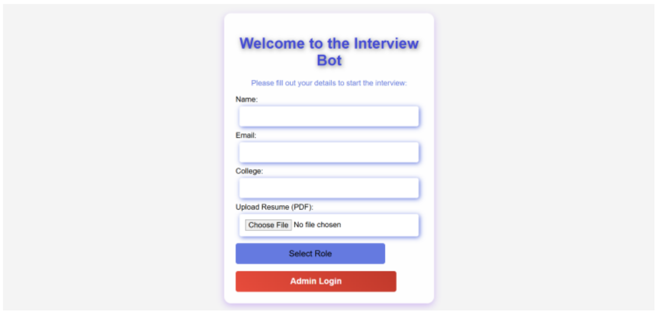
To evaluate the effectiveness and reliability of the proposed AI-enabled Interview Evaluation System, a pilot test was conducted using multiple candidate profiles. Each candidate participated in the full evaluation cycle, including registration, MCQ-based technical assessment, verbal answer submission, and face validation.

The results were assessed based on the MCQ score (technical knowledge), verbal score (communication and reasoning), and the system-generated interview outcome. Table I summarizes the performance of two sample candidates:

Candidate	MCQ Score	Verbal Score	Final Outcome

	(/10)	(/30)	
John Doe	8	22	Recommended
Jane Roe	4	10	Needs Improvement

Table I: Candidate Performance Summary


Welcome to the Interview Bot

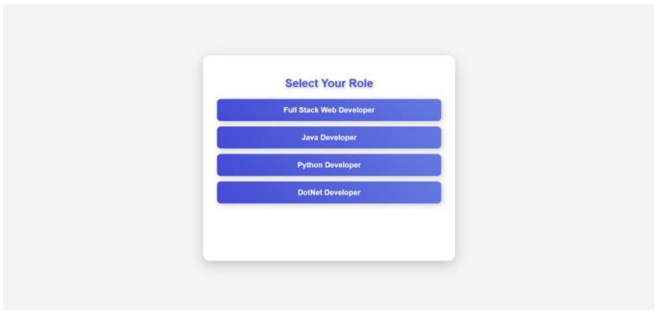
Please fill out your details to start the interview:

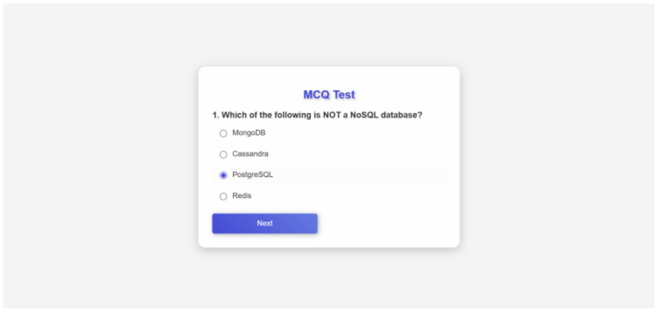
Name:

Email:

College:

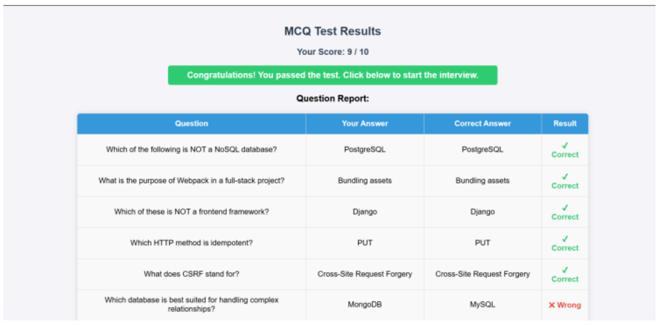
Upload Resume (PDF):


Select Your Role


MCQ Test

1. Which of the following is NOT a NoSQL database?

MongoDB
Cassandra
[✓ PostgreSQL]
Redis

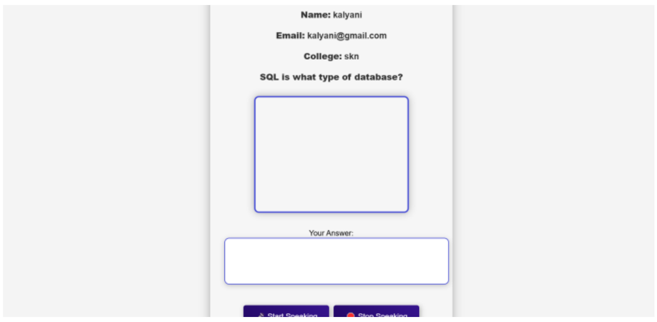

MCQ Test Results

Your Score: 9 / 10

Congratulations! You passed the test. Click below to start the interview.

Question Report:

Question	Your Answer	Correct Answer	Result
Which of the following is NOT a NoSQL database?	PostgreSQL	PostgreSQL	✓ Correct
What is the purpose of Webpack in a full-stack project?	Bundling assets	Bundling assets	✓ Correct
Which of these is NOT a frontend framework?	Django	Django	✓ Correct
Which HTTP method is idempotent?	PUT	PUT	✓ Correct
What does CSRF stand for?	Cross-Site Request Forgery	Cross-Site Request Forgery	✓ Correct
Which database is best suited for handling complex relationships?	MongoDB	MySQL	✗ Wrong

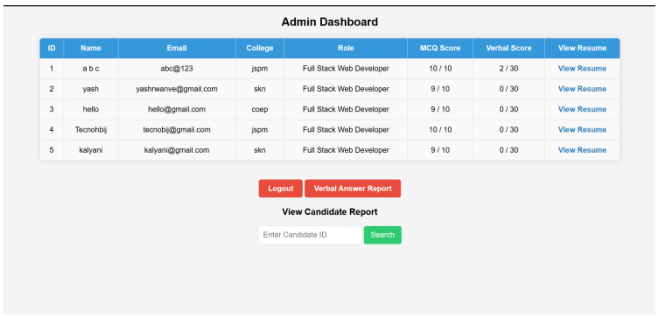

Name: kalyani

Email: kalyani@gmail.com

College: skn

SQL is what type of database?

Your Answer:



Admin Dashboard

ID	Name	Email	College	Role	MCQ Score	Verbal Score	View Resume
1	a b c	abc@123	jspm	Full Stack Web Developer	10 / 10	2 / 30	View Resume
2	yash	yashrwanve@gmail.com	skn	Full Stack Web Developer	9 / 10	0 / 30	View Resume
3	hello	hello@gmail.com	coep	Full Stack Web Developer	9 / 10	0 / 30	View Resume
4	Tecnohbij	tecnobij@gmail.com	jspm	Full Stack Web Developer	10 / 10	0 / 30	View Resume
5	kalyani	kalyani@gmail.com	skn	Full Stack Web Developer	9 / 10	0 / 30	View Resume

A. MCQ Evaluation

John Doe demonstrated a strong grasp of core technical concepts, achieving a score of 8/10 in the MCQ round.

Jane Roe, however, scored below the qualifying threshold (6/10), indicating weaker technical proficiency.

B. Verbal Answer Evaluation

The semantic scoring system, based on cosine similarity from SentenceTransformer embeddings, revealed that John Doe provided contextually rich and well-articulated answers. His high verbal score (22/30) reflected clear understanding and articulation of both technical and behavioral topics.

Jane Roe's responses, while complete in structure, lacked semantic closeness to the ideal answers. The model scored her at 10/30, which aligned with limited conceptual clarity and articulation.

This analysis supports the correlation between higher semantic similarity scores and actual communication skills, validating the use of transformer-based NLP models for verbal assessment.

C. Face Detection Accuracy

During interviews, the face validation module operated in real-time, successfully identifying:

Multiple faces in a frame, which triggered immediate alerts to candidates.

Candidate absence, when no face was detected during response recording.

These events were handled via browser alerts, enhancing interview integrity and discouraging impersonation or external interference. This validated the reliability of OpenCV’s Haar Cascade classifiers as a lightweight, real-time identity verification mechanism.

D. Overall System Outcome

The system proved successful in:

Automatically differentiating between high- and low-performing candidates

Providing structured feedback through PDF reports

Detecting and preventing unethical behavior

In summary, the results affirm the system’s ability to evaluate both technical and soft skills effectively, providing a robust solution for initial candidate screening.

VI. Future Scope

While the current implementation of the AI-enabled Interview Evaluation System successfully addresses many limitations of traditional interview processes, several enhancements can be integrated to further increase its robustness, scalability, and intelligence. The following future enhancements are proposed for the next iteration of the system:

Translatable using NLP models (e.g., MarianMT or Google Translate API)

Evaluated semantically using language-specific SentenceTransformer models

This would enable candidates to comfortably answer in their native language while still ensuring fair scoring and consistent evaluation, thereby improving accessibility and reducing language-based discrimination.

D. Recruiter Rating and Feedback Mechanism

A new feature could be added for recruiters and interviewers to provide post-evaluation feedback. This would include:

Rating candidate performance on custom rubrics

Flagging potentially misleading or AI-generated answers

Adding comments to individual question responses

Such a hybrid model—automated scoring supported by optional human review—ensures checks and balances in critical hiring decisions and facilitates better decision tracking for future audits.

A. Integration of Advanced Speech-to-Text Engines

The existing verbal module uses the browser-based Web Speech API, which is dependent on the client's browser and device capabilities. This limits transcription accuracy in noisy environments or with non-native accents. Future versions can incorporate more powerful AI-based engines like:

Whisper by OpenAI – A robust speech recognition model trained on multilingual and multitask datasets, capable of transcribing noisy, low-quality audio with high precision.

Google Cloud Speech-to-Text API – A scalable cloud-based service offering real-time transcription, speaker diarization, and automatic punctuation.

These integrations would enable more accurate transcription, especially during voice-based verbal interviews, resulting in improved NLP evaluation and fairer scoring.

B. Facial Expression and Emotion Analysis

Currently, the face detection module verifies identity based on the presence or absence of faces. Future upgrades can include facial expression recognition to assess:

Candidate stress levels

Confidence indicators

Facial cues related to dishonesty or discomfort

This can be achieved using deep learning models such as Convolutional Neural Networks (CNNs) trained on emotion-labeled facial datasets (e.g., FER2013, AffectNet). Such analysis will add an emotional intelligence layer to the interview, offering behavioral insights to complement semantic and technical evaluation.

C. Multilingual Interview Support

To make the platform inclusive for diverse linguistic groups, support for multiple languages can be integrated. Both question delivery and candidate responses should be:

By adopting these forward-looking enhancements, the system can evolve into a comprehensive, intelligent, and inclusive interview platform—capable of adapting to real-world HR needs across different domains, cultures, and languages.

VII. CONCLUSION

The increasing demand for objective, efficient, and scalable hiring processes has driven the adoption of intelligent recruitment solutions across industries. The **AI-enabled Automated Interview Evaluation System** presented in this paper offers a comprehensive approach to candidate assessment by combining multiple advanced technologies under a unified platform.

Through the use of **natural language processing**, the system evaluates the semantic relevance of verbal responses using transformer-based models, eliminating the subjectivity commonly associated with manual evaluations. The **technical MCQ test module** ensures a role-specific assessment of domain knowledge, while the **real-time face detection mechanism** helps preserve the integrity of remote interviews by identifying impersonation or unauthorized collaboration.

The backend, built using **Flask and SQLite**, ensures data persistence and processing reliability, while the frontend—designed with HTML, CSS, and JavaScript—provides a smooth and user-friendly interface. Additional features such as **speech-to-text input**, **PDF report generation**, and **admin dashboards** further enhance usability for both candidates and recruiters.

The results obtained during testing validate the system's effectiveness in differentiating high-performing candidates based on both technical competence and communication skills. The integrated design reduces human bias, minimizes administrative overhead, and enables organizations to **evaluate large numbers of applicants quickly and consistently**.

This implementation demonstrates how AI can be successfully applied to streamline one of the most resource-intensive HR functions—recruitment. By leveraging semantic similarity models, computer vision, and modular web architecture, the system sets a foundation for future development in intelligent hiring tools. It also opens pathways to incorporate deeper analytics, emotional intelligence, and language diversity into digital hiring platforms.

In conclusion, the system not only meets current industry needs but also provides a **strong and extensible base for innovation in HR technology**, particularly in scenarios requiring remote, automated, and unbiased candidate evaluation.

REFERENCES

[1] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," *arXiv preprint arXiv:1908.10084*, 2019.
[2] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv preprint arXiv:1810.04805*, 2018.
[3] OpenCV, "Open Source Computer Vision Library," [Online]. Available: https://opencv.org/.
[4] F. Chollet, "Deep Learning with Python," Manning Publications, 2017.
[5] M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," *arXiv preprint arXiv:1603.04467*, 2016.
[6] A. Vaswani et al., "Attention is All You Need," in *Proc. Advances in Neural Information Processing Systems*, 2017.
[7] M. C. Mozer, R. Wolniewicz, D. B. Johnson, and R. E. Harrington, "Predicting the Performance of Human Job Applicants," *Int. J. of Forecasting*, vol. 17, no. 3, pp. 383–398, 2001.
[8] Flask Documentation, "The Pallets Projects," [Online]. Available: https://flask.palletsprojects.com/.
[9] SQLite Consortium, "SQLite: Lightweight SQL Database Engine," [Online]. Available: https://www.sqlite.org/.
[10] T. Mikolov et al., "Efficient Estimation of Word Representations in Vector Space," *arXiv preprint arXiv:1301.3781*, 2013.
[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Proc. NIPS*, 2012.

[12] FPDF Documentation, “Free PDF Generation Library for Python,” [Online]. Available: http://www.fpdf.org.
[13] Mozilla Developer Network, “Web Speech API,” [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API.
[14] R. Caruana and A. Niculescu-Mizil, “An Empirical Comparison of Supervised Learning Algorithms,” in *Proc. ICML*, 2006.
[15] Google Cloud, “Speech-to-Text API,” [Online]. Available: https://cloud.google.com/speech-to-text.
[16] OpenAI, “Whisper: Speech Recognition Model,” [Online]. Available: https://github.com/openai/whisper.
[17] A. Mollahosseini, D. Chan, and M. H. Mahoor, “Going Deeper in Facial Expression Recognition using Deep Neural Networks,” *Proc. IEEE Winter Conf. on Applications of Computer Vision (WACV)*, 2016.
[18] R. Picard, “Affective Computing,” *MIT Press*, 1997.
[19] UKP Lab, “Sentence Transformers,” [Online]. Available: https://www.sbert.net/.
[20] R. E. Schapire, “The Strength of Weak Learnability,” *Machine Learning*, vol. 5, no. 2, pp. 197–227, 1990.