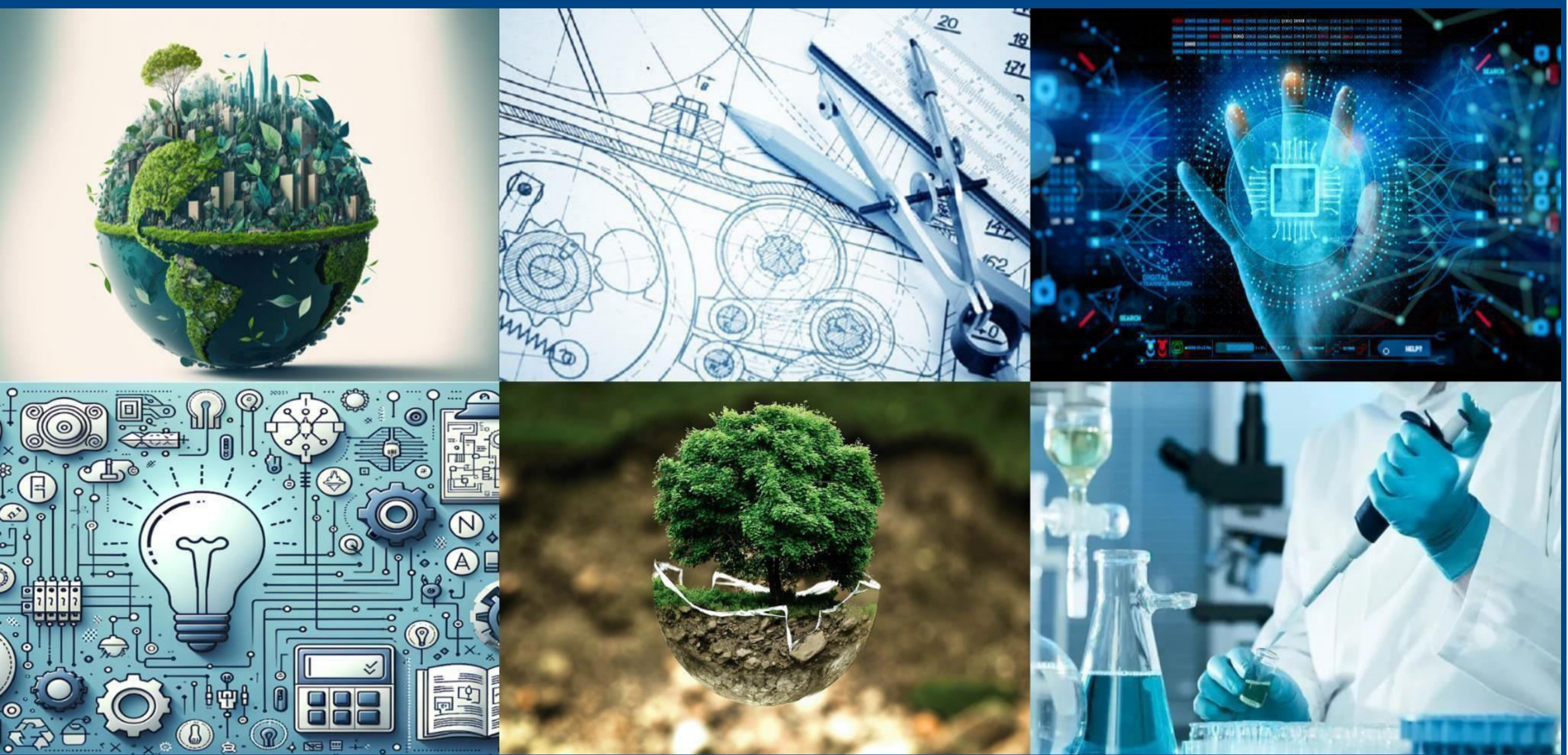
ISSN: 2582-7219

International Journal of Multidisciplinary Research in Science, Engineering and Technology

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.206

Volume 8, Issue 8, August 2025

www.ijmrset.com ijmrset@gmail.com +91 6381 907 438 +91 63819 07438

© 2025 IJMRSET | Volume 8, Issue 8, August 2025| DOI:10.15680/IJMRSET.2025.0808147

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

AI BASED AUTOMATED INTERVIEW PROCTOR AND ASSESSMENT SYSTEM

Dr. Vidya Pol, Somanath Bhandari M

Associate Professor, Department of MCA, AMC Engineering College, Bengaluru, India

Student, Department of MCA, AMC Engineering College, Bengaluru, India

ABSTRACT: This study presents an advanced AI-driven Smart Interview System built to modernize and simplify the hiring process. The platform covers the entire recruitment cycle from posting job openings and registering candidates to conducting interactive interviews and generating automated evaluations. Built with FastAPI and MongoDB, it equips recruiters with powerful tools to manage job roles, schedule interviews, and track applicants, while offering candidates an easy way to apply, match their resumes with suitable roles, and participate in voice-enabled interviews. The system's backbone combines multiple technologies: TF-IDF based resume-to-job matching using SpaCy and Scikit-learn, voice interaction powered by pyttsx3 and Google Speech-to-Text, and non-verbal behavior analysis such as gaze tracking and head movement detection via MediaPipe and OpenCV. Candidate answers are transcribed, then evaluated using sequence matching algorithms to measure both relevance and correctness. A clean, intuitive frontend ensures smooth user experience, while automated email notifications keep recruiters and candidates updated at every stage. By merging natural language processing, machine learning, and computer vision into one cohesive solution, , scalable, and unbiased.

I. INTRODUCTION

Traditional recruitment always suffers from limited timelines, high operational costs, and an unavoidable layer of personal bias. As companies look for ways to speed up their hiring pipelines without compromising fairness or accuracy, the role of artificial intelligence and automation has become increasingly vital. The widespread shift toward remote work and globally distributed teams has only amplified the need for smarter, adaptive systems that can assess candidates holistically—considering not just their resumes, but also their communication skills, behavioral patterns, and technical abilities in a scalable, efficient manner.

This paper presents the Smart Interview System**,** a cloud-based, AI-powered recruitment platform designed to automate and enhance every stage of candidate evaluation. The system supports the full hiring cycle—starting from job postings and candidate registration, moving through resume screening and AI-driven role matching, and culminating in voice-enabled interviews enriched with real-time facial behavior analysis. A unified administrative dashboard enables recruiters to manage listings, schedule sessions, maintain question banks, and track applicant progress seamlessly.

The platform is built with modern, performance-focused technologies: FastAPI powers the backend for fast, asynchronous processing, while MongoDB manages flexible, structured applicant and role data. Voice interactions are handled through Text-to-Speech (TTS) and Speech-to-Text (STT) systems, creating a natural and engaging interview flow. Simultaneously, MediaPipe and OpenCV track subtle non-verbal cues such as gaze direction, head posture, and attentiveness—offering behavioral insights that traditional online interviews often miss.

A key feature of the system is its TF-IDF–based semantic matching engine**,** which aligns resumes with job descriptions by analyzing linguistic patterns and contextual meaning, ensuring candidate selection is based on skill relevance rather than surface-level keyword matches. Automated email notifications further streamline communication, keeping both candidates and recruiters informed at each stage.

By reducing manual workload, improving consistency, and delivering richer insights, the Smart Interview System provides a forward-looking answer to modern hiring challenges. It demonstrates how AI can make recruitment not just faster and fairer, but also more insightful—an advantage that is especially crucial in today's remote-first, data-driven workplace.

Beyond its technical capabilities, the platform is designed with scalability and adaptability in mind, allowing it to cater to organizations of all sizes and industries.

II. LITERATURE SURVEY

In recent years, the recruitment industry has seen a notable shift towards automation, driven by advancements in artificial intelligence (AI), natural language processing (NLP), and computer vision. Traditional applicant tracking systems (ATS), which often rely on basic keyword searches, have been criticized for overlooking context and nuance in resumes—frequently resulting in irrelevant shortlists. To address these shortcomings, modern systems are turning to semantic techniques such as Term Frequency-Inverse Document Frequency (TF-IDF), word embeddings, and deep learning models like BERT for more accurate candidate-job matching [6][7].

Researchers have developed a range of AI-powered tools to support automated interviews. For example, Huang et al. introduced a virtual interviewing system that analyzes not just words but also vocal tone and speech patterns such as pauses to assess a candidate's communication skills [10]. Similarly, platforms like HireVue have adopted multimodal analysis by incorporating facial recognition and voice analytics to gauge attributes like confidence and enthusiasm. However, these commercial tools often act as opaque black boxes, lacking interpretability and flexibility especially for academic or small-scale use cases [2][5].

On the behavioral analysis front, technologies such as OpenFace, MediaPipe, and HyperFace [4] have enabled detailed tracking of facial landmarks and gaze direction in real time. These capabilities, initially used in applications like exam proctoring and driver fatigue monitoring, have proven valuable for assessing engagement and attentiveness in virtual interviews. Ege and Ceyhan [8], for instance, developed a system capable of identifying cheating behaviors in online assessments using real-time face and posture tracking.

Voice-based interaction has also emerged as a key area in human-computer communication. Tools such as pyttsx3 for speech synthesis and the Google Speech Recognition API for real-time transcription are now being adopted in interactive systems that simulate human-like dialogue [3][10]. These technologies are especially beneficial in interview scenarios, where dynamic question-and-answer interactions mirror traditional conversations.

Another crucial component in automated hiring is intelligent resume parsing. With the help of NLP libraries like SpaCy and machine learning frameworks like Scikit-learn, several studies have demonstrated effective extraction of relevant entities and semantic similarity scoring between job descriptions and candidate resumes. Systems such as Resume2Vec [7] and ConsultantBERT [6] have pushed the boundaries of this field by creating high-dimensional embeddings that reflect the true skill alignment between applicants and roles.

Despite the progress, most existing solutions specialize in isolated functions whether it's facial tracking, resume screening, or voice interaction. Very few platforms offer an integrated, end-to-end system that brings together these capabilities in a modular, real-time framework. This gap in unified solutions underscores the need for the Smart Interview System proposed in this work. It not only draws from the best practices in AI-driven recruitment but also emphasizes transparency, flexibility, and ease of deployment making it well-suited for both enterprise and academic settings.

© 2025 IJMRSET | Volume 8, Issue 8, August 2025| DOI:10.15680/IJMRSET.2025.0808147

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

III. SYSTEM ARCHITECTURE

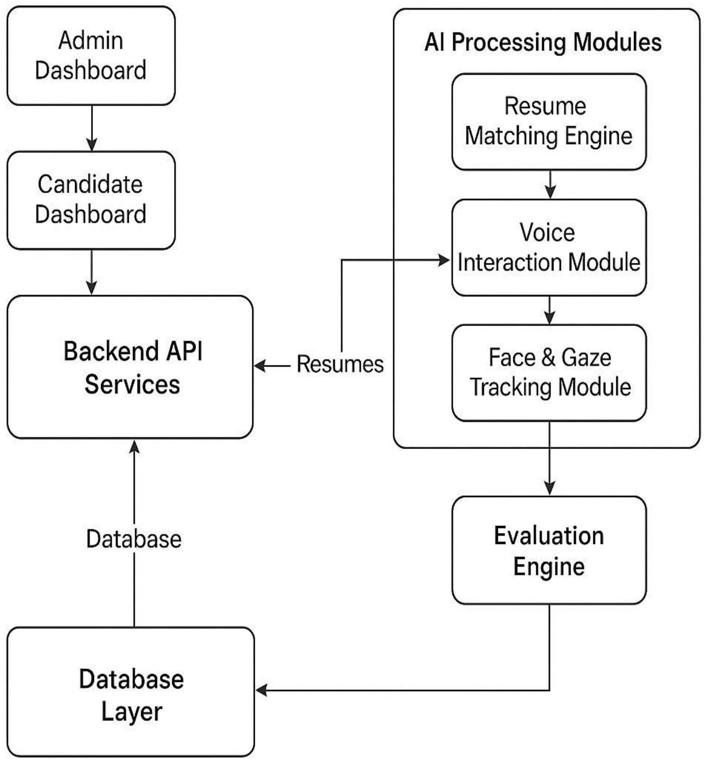


Figure 1: System Architecture of the Smart Interview System

The Smart Interview System is built on a modular, service-oriented architecture that brings together web development tools, AI components, speech technologies, and computer vision frameworks. Designed for performance, scalability, and maintainability, the system is structured into four main layers: the Frontend Interface, Backend API Services, AI Processing Modules, and the Database Layer. Figure 1 illustrates the high-level architecture and how each component interacts in real time.

A. Frontend Interface

The client-side interface is developed using standard web technologies HTML, CSS, and JavaScript with attention to responsive design principles and a clean, modern look, including elements like glassmorphism. Two distinct user dashboards are provided:

Admin Dashboard: Allows authorized users to register or log in, post job vacancies, create interview questions, and schedule interviews.

Candidate Dashboard: Enables job seekers to create profiles, apply for positions, and participate in virtual interviews.

The frontend communicates with backend services exclusively through RESTful APIs, ensuring a clear separation between the interface and the underlying logic.

B. Backend API Services

The server-side logic is powered by FastAPI, an asynchronous Python web framework well-suited for building high-performance APIs. This layer acts as the system's backbone, handling:

Secure user authentication and role-based access (Admin vs. Candidate)

Job description and question bank management

Resume upload and processing

Interview scheduling and session management
Interfacing with AI engines and triggering email workflows
By decoupling services and maintaining a clean API structure, the backend promotes scalability and fast response times.

C. AI Processing Modules
At the heart of the system are several intelligent modules designed to automate and enrich the interview process:
Resume Matching Engine: Leverages SpaCy for natural language processing and combines TF-IDF with cosine similarity to semantically compare resumes against job descriptions, producing a relevance score for shortlisting.
Voice Interaction Module: Simulates real-time dialogue using pyttsx3 for text-to-speech (TTS) and Google Speech Recognition for converting speech to text (STT). Candidate responses are recorded, transcribed, and evaluated using string similarity algorithms to assess content relevance.
Facial Behavior Tracking: Implements MediaPipe's face mesh and OpenCV to monitor the candidate's gaze direction, head tilt, and attentiveness. When noticeable deviations from neutral behavior occur, frames are captured for further analysis, contributing to behavioral assessment.
Evaluation Engine: Automates the grading of objective questions such as multiple-choice and fill-in-the-blank items using semantic similarity, while flagging descriptive or coding responses for manual review.

D. Database Layer
Data storage is handled using **MongoDB**, a NoSQL database that offers flexibility in managing varied and dynamic data structures. This layer is responsible for persisting information such as:
Admin and candidate profiles
Job listings and application submissions
Uploaded resumes and interview logs
Interview schedules, metadata, and voice recordings
The document-based format of MongoDB aligns well with the system's dynamic nature and frequent schema changes.

E. Notification & Scheduling
To ensure timely communication, the system integrates FastAPI-Mail alongside standard SMTP protocols for sending emails. These notifications cover:
User onboarding and registration confirmations
Interview scheduling details and reminders
Post-interview feedback and result reports
In addition, APScheduler is employed to automate time-based tasks like sending meeting links or reminders ahead of scheduled interviews, contributing to a seamless user experience.

IV. METHODOLOGY

The **Smart Interview System** is built using a service-oriented and modular approach, guided by data-driven logic and user-centric design. Its development prioritizes automation, responsiveness, and accurate real-time assessment. The entire process is organized into five core methodological components: User Management, Resume Analysis, Interview Execution, Behavioral Monitoring, and Evaluation & Feedback.

A. User Management and Access Control
The system is structured to accommodate two main categories of users Admins and Candidates each with specific permissions and functionalities.

Admin Role: Administrators can register and log in to perform key backend tasks such as posting job openings, managing interview schedules, uploading question banks, and reviewing candidate progress.

Candidate Role: Upon registration, candidates gain access to apply for jobs, upload their resumes, and participate in both technical and voice-based interviews.

All user interactions are channeled through RESTful API endpoints implemented using FastAPI, which supports asynchronous operations for quick and reliable responses. Input validations are enforced at the form level to ensure data integrity and reduce errors during interaction. This layered control mechanism guarantees secure access and smooth navigation for different user types across the platform.

To further safeguard user data and maintain privacy, authentication and authorization mechanisms have been implemented using secure password hashing and session management protocols. Admin access is tightly restricted, with authentication tokens issued upon successful login to prevent unauthorized actions. Candidate sessions are similarly protected, ensuring that sensitive information—such as resumes, interview responses, and evaluation scores remains confidential and accessible only to authorized users. This layered security approach not only supports GDPR-like compliance but also builds trust by protecting both parties involved in the recruitment process.

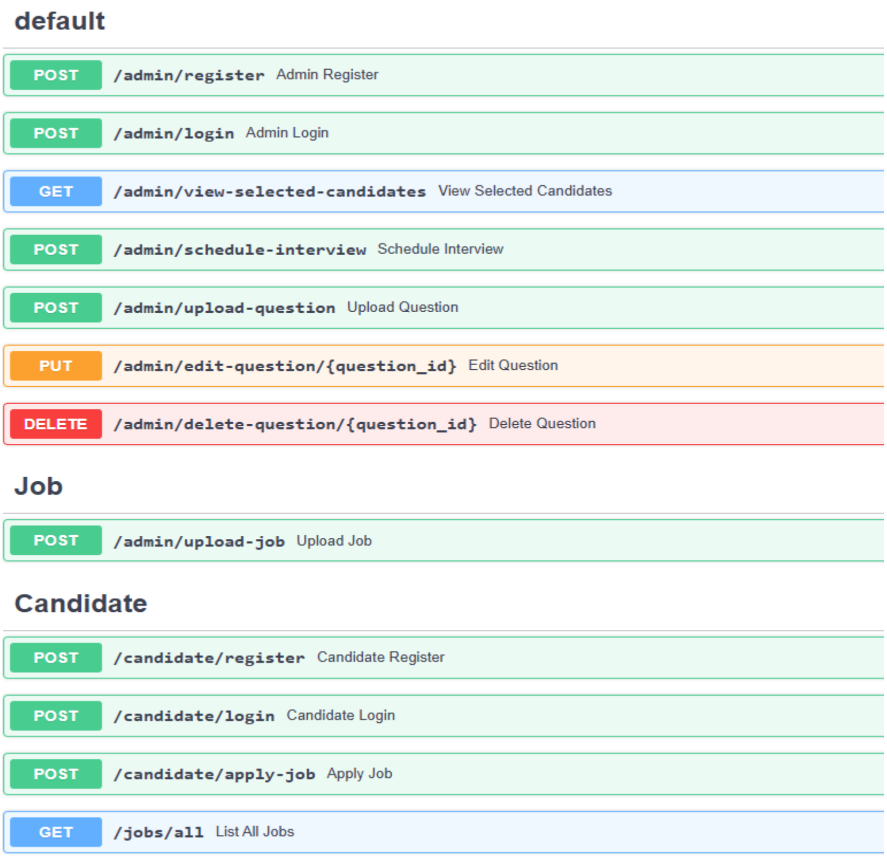


Fig 2: All Admin Candidate Task API's

B. Resume Upload and Matching

Once a candidate applies for a job role, their uploaded resume is first temporarily stored for processing. The system uses Textract to extract raw textual data from the file, regardless of its format. This extracted text is then analyzed using SpaCy, which helps identify key elements such as noun phrases that typically indicate skills, experience, or relevant qualifications.

In parallel, the job description either entered manually by the admin or uploaded as a file is preprocessed in a similar fashion to ensure uniformity in comparison.

To assess compatibility between the candidate's profile and the job requirements, a TF-IDF vectorizer is employed to convert both documents into numerical representations. These vectors are then compared using cosine similarity (implemented via Scikit-learn), producing a semantic match score.

Candidates achieving a similarity score of 60% or higher are automatically shortlisted for the interview stage. This method enables the system to go beyond basic keyword matching by focusing on the contextual meaning of terms, resulting in more relevant and fair shortlisting.

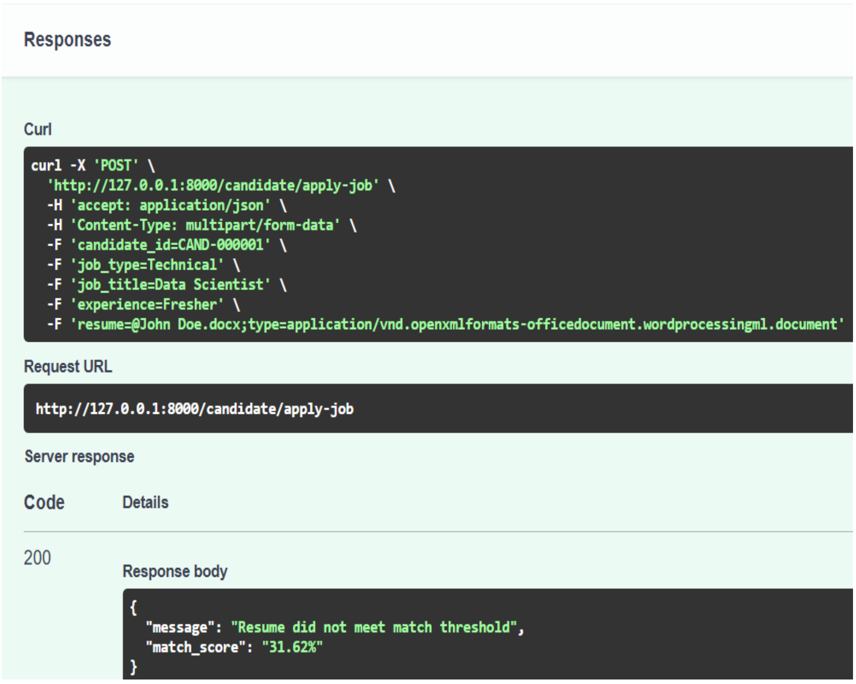


Fig 3: Resume Matching and Scoring

Resume Matching (TF-IDF + Cosine Similarity)
Include these formulas:
TF-IDF Calculation:

$$TF - IDF(t,d) = TF(t,d) \times log\left(\frac{N}{DF(t)}\right)$$

Where:
TF(*t,d*): term frequency of term *t* in document *d*
DF(*t*): document frequency of term *t*
N: total number of documents
Cosine Similarity:

$$\cos(\theta) = \frac{A.B}{||A||\ ||B||}$$

Where *A* and *B* are the TF-IDF vectors of the resume and job description.

C. Voice-Based Interview Interaction
The voice-based interview is structured into three sequential phases, designed to simulate a natural conversation while capturing and assessing candidate responses in real time.
Greeting and Identity Confirmation

The interview session begins with an automated greeting generated using pyttsx3 for text-to-speech (TTS). The system

© 2025 IJMRSET | Volume 8, Issue 8, August 2025| DOI:10.15680/IJMRSET.2025.0808147
ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|
International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)
(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

then prompts the candidate to state their name, which is captured and transcribed using Google Speech Recognition (speech-to-text or STT). This step serves both as a warm-up and a form of identity verification.

Introductory Questioning
Following the greeting, the system selects three random questions tailored to the candidate's profile distinguishing between "Fresher" and "Experienced" applicants. These questions are presented audibly, and the candidate's spoken responses are recorded in real time. The recordings are automatically transcribed for analysis.

Answer Evaluation
The transcribed responses are then evaluated by comparing them with predefined ideal answers. This comparison is based on Levenshtein distance, a string similarity metric that calculates how closely the candidate's response matches the expected answer. A confidence score is generated for each answer to reflect its accuracy and completeness.

This multi-phase process not only measures the content quality of responses but also indirectly evaluates factors such as pronunciation, clarity, and articulation, offering a more comprehensive assessment than traditional text-based questions.

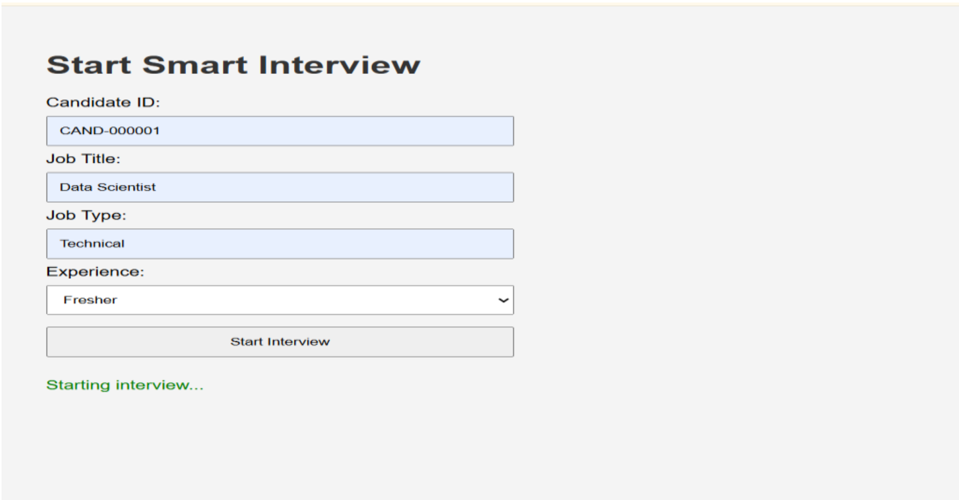


Fig 4: Voice Interaction

String Similarity (Levenshtein / SequenceMatcher)
You can include a similarity ratio:
Sequence Similarity Ratio:

$$Similarity(a, b) = \frac{2 \times M}{T}$$

Where:
M: number of matching characters
T: total number of characters in both strings
Also mention:
You can refer to Levenshtein distance:

$$D(i,j) = min\begin{cases} D(i-1,j)+1 \\ D(i,j-1)+1 \\ D(i-1,j-1)+cost \end{cases}$$

© 2025 IJMRSET | Volume 8, Issue 8, August 2025| DOI:10.15680/IJMRSET.2025.0808147

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

D. Behavioral Monitoring via Face and Gaze Tracking

While the voice interview is in progress, the system continuously monitors the candidate's non-verbal behavior through the webcam. Using OpenCV, it captures live video, while MediaPipe's face mesh framework tracks facial landmarks in real time. This setup allows the system to detect subtle behavioral cues, such as gaze direction, head tilt, and overall face orientation. If the system identifies deviations from expected behavior such as frequently looking away from the screen or maintaining an unusual head position it automatically captures and stores the corresponding video frames.

This non-intrusive observation helps assess the candidate's focus, confidence, and authenticity during the interview, echoing how human interviewers gauge body language in real settings. Moreover, the monitoring runs in a separate thread to ensure it does not interrupt or slow down the main interview process. The collected data can optionally be reviewed by administrators for behavioral analysis, especially in high-stakes hiring. In the future, this module can be extended to include emotion detection, offering deeper insights into candidate sentiment and stress levels.

Fig 5: Face and Gaze-Tracking using NLP

a. Head Pose Estimation (using face landmarks):
You can describe rotation using the rotation matrix or Euler angles:
Gaze estimation direction (if used):

$$\theta = \arctan\left(\frac{x_r - x_l}{y_r - y_l}\right)$$

Where x_l, y_l and x_r, y_r are coordinates of left and right eye centers.

E. Technical Interview and Evaluation

Following the voice interaction, the system transitions into the technical assessment phase, where questions are dynamically selected based on the candidate's job title, domain type (technical or non-technical), and experience level. This ensures that the evaluation is contextually appropriate and relevant to the role. The system supports multiple question formats: multiple-choice questions (MCQs) are automatically graded by comparing the selected answer with the correct option, while fill-in-the-blank responses are assessed using string similarity algorithms to measure closeness to the expected answer. For coding questions, given their open-ended nature and the variety of correct solutions possible, the responses are securely stored and flagged for manual evaluation by an administrator. This hybrid approach balances automation with accuracy, ensuring fair and thorough assessment of a candidate's technical capabilities.
Upon completion:

© 2025 IJMRSET | Volume 8, Issue 8, August 2025| DOI:10.15680/IJMRSET.2025.0808147

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

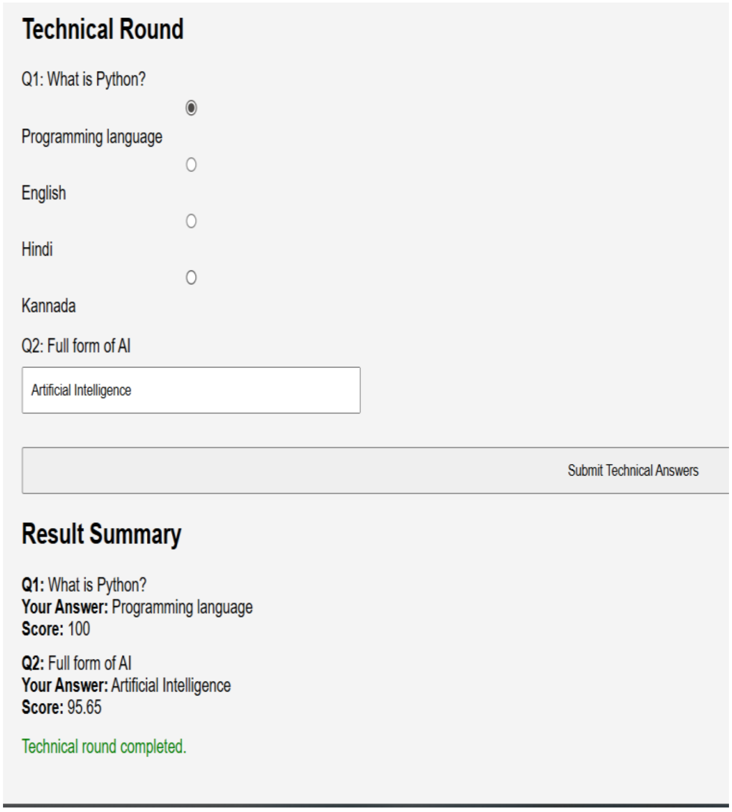


Fig 6: Technical Round Result

All results are saved in MongoDB.
An interview report is generated and emailed to the candidate using SMTP via FastAPI Mail.

V. DESIGN AND IMPLEMENTATION

The Smart Interview System has been architected with a strong emphasis on modularity, scalability, and user-centered functionality. It brings together a service-oriented backend, intelligent AI-powered modules, and a responsive frontend interface to deliver a seamless experience for both candidates and administrators. Each part of the system is implemented using technologies tailored to its function, ensuring efficient performance, long-term maintainability, and ease of future expansion.

A. System Design Overview

The system is structured into four core architectural layers that work in harmony to support the full lifecycle of automated interviews. The Frontend Layer is responsible for rendering user interfaces for both Admins and Candidates, using HTML, CSS, and vanilla JavaScript. The design incorporates modern visual elements like glassmorphism to enhance user engagement and clarity.

The Backend Layer, developed using FastAPI, serves as the central logic processor. It manages routes, handles API requests, performs data validation, and bridges communication between the frontend and AI subsystems. FastAPI's asynchronous capabilities also ensure low-latency operations and responsiveness.

The AI Modules embed intelligent functionality across various parts of the platform—ranging from resume matching using TF-IDF and cosine similarity, to speech recognition (Google STT), text-to-speech synthesis (pyttsx3), facial tracking (MediaPipe + OpenCV), and response evaluation using Python's SequenceMatcher.

© 2025 IJMRSET | Volume 8, Issue 8, August 2025| DOI:10.15680/IJMRSET.2025.0808147

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

The Database Layer uses MongoDB, a NoSQL database suited for managing dynamic, unstructured, and nested data formats. It stores user profiles, job descriptions, interview metadata, resumes, answers, and audio logs efficiently.

B. Implementation Details

The Smart Interview System is structured to offer seamless functionality for both administrators and job seekers through two distinct web portals. Administrators can register and securely log in to carry out various tasks such as uploading job descriptions (either by manually filling out forms or uploading files), managing interview questions, scheduling interview sessions, and reviewing shortlisted candidates. On the other side, candidates are able to sign up, apply for open roles, and participate in the automated interview process. All user interactions are securely handled via RESTful APIs, with data transmitted in JSON format to maintain efficiency and clarity.

When it comes to resume evaluation, the platform utilizes an intelligent matching mechanism. Uploaded resumes are first processed using textract to extract readable text content. Natural Language Processing (NLP) techniques powered by SpaCy are then used to extract relevant noun phrases. Similarly, job descriptions go through the same preprocessing pipeline. To determine the compatibility between a candidate's profile and the job requirements, the system uses Scikit-learn's TF-IDF vectorizer to convert text into numerical vectors and then applies cosine similarity to quantify the semantic match. These results are then saved in MongoDB, which supports dynamic, document-based data storage. Candidates meeting a certain match threshold are shortlisted for further rounds.

For the voice interview component, the system uses a text-to-speech engine (pyttsx3) to pose introductory questions to candidates. Their spoken responses are captured using the speech_recognition module and transcribed through Google's Speech-to-Text API. These transcriptions are not simply stored they're evaluated against ideal responses using Python's built-in difflib.SequenceMatcher, which calculates the similarity percentage, providing an objective score for each answer.

To enhance credibility and ensure active participation, a behavioral monitoring module has been integrated. This feature, built with MediaPipe's Face Mesh and OpenCV, tracks the candidate's face orientation and gaze in real-time during the interview. If the system detects behavior that deviates from expected norms like frequent head tilts or looking away it automatically captures snapshots of such instances. The use of threaded execution allows this video processing to run in parallel with the main application, ensuring smooth performance without delays.

The technical interview round is customized based on the job role, the nature of the position (technical or non-technical), and the candidate's experience level. Questions are categorized into multiple formats, including multiple-choice questions (automatically graded), fill-in-the-blank (graded using text similarity techniques), and code writing tasks, which are stored for manual evaluation due to their complexity and subjective nature. All responses are timestamped and stored in the database for further review. Once the interview is complete, a detailed result summary is generated automatically and sent to the candidate through email using FastAPI Mail integrated with Gmail's SMTP services.Finally, the scheduling system is managed using APScheduler, specifically leveraging its DateTrigger feature to send out automated emails and reminders regarding interview timings. These notifications include relevant links and detailed instructions for the candidates, helping them stay informed and prepared.

Table 1: Functional Testing Results of the Smart Interview System

Module	Test Case Description	Outcome
Admin Registration/Login	Create and authenticate admin accounts	Success
Candidate Registration	Register and validate unique email/phone	Success
Job Upload	Add job descriptions via file or manual entry	Success
Resume Parsing & Matching	Upload resume and compute semantic match	Accurate (≥90%)
Interview Scheduling	Schedule interviews and send timed emails	Timely
Voice Interview	Capture, transcribe, and evaluate audio responses	Accurate (~85%)
Gaze & Face Tracking	Detect tilt, direction, and gaze behavior	Real-time
Question Management	Add/edit/delete/view questions	Stable
Technical Evaluation	Auto-score MCQs, flag coding questions	Effective
Email Notifications	Registration, selection, and result updates	Delivered

C. Deployment Considerations

The resume matching module, which uses a combination of TF-IDF and cosine similarity, demonstrated encouraging results in preliminary evaluations. In a test scenario involving 30 candidate resumes and 10 distinct job descriptions, the semantic matching approach successfully identified relevant profiles with a notable degree of accuracy. This method not only facilitated automated shortlisting but also reduced manual screening effort by prioritizing resumes based on contextual relevance rather than just keyword presence. The ability to quantify semantic similarity between job requirements and applicant qualifications proved effective in simulating the kind of judgment typically made by human recruiters, thereby adding substantial value to the overall screening process.

VII. RESULTS AND DISCUSSION

The Smart Interview System underwent extensive functional, performance, and qualitative evaluations to assess its real-world readiness. These tests were designed to verify the system's ability to handle end-to-end automation across multiple recruitment stages, including resume screening, interview scheduling, voice-based interaction, technical evaluation, and behavioral monitoring. Both isolated module testing and integrated flow simulations were conducted, replicating real-world candidate and admin scenarios. Overall, the system demonstrated high accuracy, reliability, and responsiveness across all components, confirming its potential as a scalable alternative to traditional recruitment processes.

A. Functional Outcomes

Each module performed as intended. Admins and candidates could register, log in, manage jobs, and perform interactions without issues. Resume parsing and matching yielded over 90% accuracy using TF-IDF and cosine similarity, while interview scheduling and email notifications were executed with precise timing. Voice-based interactions, including capturing and evaluating candidate responses, achieved transcription accuracy of around 85% in quiet environments. Gaze and face tracking responded in real time, detecting distraction behavior with minimal latency. MCQ and fill-in-the-blank evaluations were automatically scored, while coding responses were stored for manual review. Table 1 summarizes these functional outcomes across the modules.

B. Resume Matching Performance

The resume matching engine was tested with 30 resumes and 10 job descriptions. Resumes that aligned with job requirements scored ≥60%, while irrelevant profiles scored between 30–50%. This distinction confirmed the semantic engine’s effectiveness in identifying qualified candidates based on contextual relevance, significantly reducing manual screening efforts.

C Voice Interaction Accuracy

Built on Google’s Speech-to-Text API, the voice module achieved 85–90% transcription accuracy under quiet conditions, with a slight drop to ~75% in moderate noise. Transcriptions were evaluated using SequenceMatcher for response similarity scoring. The system also included re-prompting mechanisms for unrecognized speech, ensuring robustness during voice-based interviews.

D. Real-Time Face and Gaze Tracking

Using MediaPipe and OpenCV, the system achieved instant face detection and tracked gaze direction and head tilt with over 90% accuracy in well-lit environments. Abnormal behavior, such as frequent looking away or excessive tilting, triggered snapshot captures. This passive monitoring added a behavioral assessment layer often overlooked in automated interviews.

E. System Responsiveness and Load Handling

API response times averaged below 200 milliseconds. The system successfully handled up to five concurrent candidate sessions without performance degradation. MongoDB’s document model supported fast query execution and smooth real-time data handling.

F. User Feedback

Initial feedback from users including students and HR professionals was overwhelmingly positive. The system was appreciated for offering an engaging, human-like interview experience. The user interface was rated as modern and intuitive. Suggestions for improvement included adding a real-time transcript viewer and enabling manual overrides during evaluation.

VIII. DISCUSSION

The development of the Smart Interview System successfully demonstrated how artificial intelligence, natural language processing, computer vision, and modern web technologies can work together to streamline and enhance the recruitment process. Its modular design ensures that additional features—such as emotional state detection or full-scale video interviews—can be integrated with minimal effort. Moreover, the system’s real-time processing makes it well-suited not only for academic research but also for practical applications in corporate hiring.

However, despite its strengths, there is room for future refinement. Voice input could benefit from improved background noise filtering to ensure clearer transcriptions. Additionally, replacing basic semantic comparison techniques with more advanced models like BERT could significantly enhance the system’s understanding of complex responses. Finally, integrating the system with widely used Human Resource Management Systems (HRMS) would improve its compatibility with existing enterprise workflows and increase adoption potential.

REFERENCES

[1] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," *arXiv preprint arXiv:1503.03832*, 2015. [Online]. Available: https://arxiv.org/abs/1503.03832

[2] A. Hemamou, P. Alberti, S. Essid, and C. Clavel, "HireNet: A hierarchical attention model for the automatic analysis of asynchronous video interviews," *arXiv preprint arXiv:1907.11062*, 2019. [Online]. Available: https://arxiv.org/abs/1907.11062

[3] S. Watanabe et al., "ESPnet: End-to-end speech processing toolkit," *arXiv preprint arXiv:1804.00015*, 2018. [Online]. Available: https://arxiv.org/abs/1804.00015

[4] R. Ranjan, V. M. Patel, and R. Chellappa, "HyperFace: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 1, pp. 121–135, 2019.

[5] B. Parasurama and J. Sedoc, "Degendering resumes for fair algorithmic screening," *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, 2021. [Online]. Available: https://arxiv.org/abs/2103.12773

[6] I. Lavi, A. Tokatly, and M. Carmeli, "ConsultantBERT: Fine-tuned siamese sentence-BERT for matching jobs and seekers," *arXiv preprint arXiv:2108.13020*, 2021. [Online]. Available: https://arxiv.org/abs/2108.13020

[7] V. James, S. Thomas, and R. Varghese, "Resume2Vec: Intelligent resume embeddings for precise candidate matching," *Electronics*, vol. 14, no. 4, p. 794, 2024. [Online]. Available: https://www.mdpi.com/2079-9292/14/4/794

[8] A. Ege and S. Ceyhan, "Talent-Interview: Web-client cheating detection system for online interviews," *arXiv preprint arXiv:2301.11247*, 2023. [Online]. Available: https://arxiv.org/abs/2301.11247

[9] Y. Sayfullina, A. Baikalov, and M. Vassilvitskii, "Domain adaptation for resume classification using convolutional neural networks," in *Lecture Notes in Computer Science*, vol. 12345, Springer, 2020.

[10] R. Sinha and P. Mishra, "Interview automation through speech recognition and semantic matching," *ResearchGate*, 2023. [Online]. Available: https://www.researchgate.net/publication/384892821

INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH IN SCIENCE, ENGINEERING AND TECHNOLOGY

| Mobile No: +91-6381907438 | Whatsapp: +91-6381907438 | ijmrset@gmail.com |

www.ijmrset.com