Automated AI System For Interview Fraud Analysis And Alerts

B. Nirmal[1], P. Abinesh[2], A. Manikandan[3], Mrs. K. Menaka[4]
[1, 2, 3]Dept of Computer science and Engineering
[4]Assistant Professor, Dept of Computer science and Engineering
[1, 2, 3, 4] Sree Sowdambika College of Engineering

Abstract- *The rapid shift to remote hiring has created new avenues for interview fraud. This paper presents an Automated AI System for Interview Fraud Analysis and Alerts, a browser-native web application that monitors online interview sessions in real time using a multi-modal detection pipeline. The system integrates computer vision via MediaPipe Face Mesh and TensorFlow.js COCO-SSD for face monitoring, eye gaze estimation, and object detection; Web Audio API for multi-voice and whisper detection; a QR code-based mobile device pairing module backed by Firebase Realtime Database; and browser tab-switch detection using the Page Visibility API. All violations are mapped to a dynamic integrity score beginning at 100 points. An administrator dashboard provides post-session review with candidate-level integrity scores and violation histories. The system extends prior work on audio-visual synchronization-based fraud detection by delivering proactive, real-time detection across six independent modalities entirely within the candidate's browser.*

Keywords: Interview Fraud Detection, Computer Vision, Online Proctoring, MediaPipe, TensorFlow.js, Firebase, Integrity Monitoring, Multi-Voice Detection, Web Application

I. INTRODUCTION

The global shift toward remote hiring practices has significantly transformed recruitment workflows. Online video interview platforms such as Google Meet, Zoom, and Microsoft Teams have become the standard medium through which candidates are evaluated. While this shift has improved accessibility and efficiency, it has also introduced new forms of candidate misconduct including the use of unauthorized reference materials, proxy candidates, pre-recorded audio playback, and multi-device coordination strategies.

Existing proctoring solutions are predominantly designed for academic examination environments and often require dedicated software installations, intrusive system-level permissions, or proprietary hardware. These solutions are rarely tailored to the specific dynamics of professional interview settings, which demand a lighter, more adaptive approach.

This paper presents an Automated AI System for Interview Fraud Analysis and Alerts — a browser-native web application built using JavaScript, HTML5, and CSS3. The system employs a six-module detection pipeline that operates entirely within the candidate's browser using MediaPipe, TensorFlow.js, and the Web Audio API, backed by Firebase Realtime Database for cross-device communication and session persistence. The system is directly inspired by and extends the work of Ramana Babu et al. [1], which proposed post-interview fraud detection via audio-visual synchronization, by delivering equivalent capabilities in real time during live sessions.

The primary contributions are: (1) a multi-modal real-time detection pipeline covering six fraud modalities; (2) a QR code-based mobile device pairing system for physical movement tracking; (3) a dynamic integrity scoring mechanism; and (4) an administrator dashboard for post-session review. The system is live and accessible at: https://heartfelt-torrone-6e9673.netlify.app

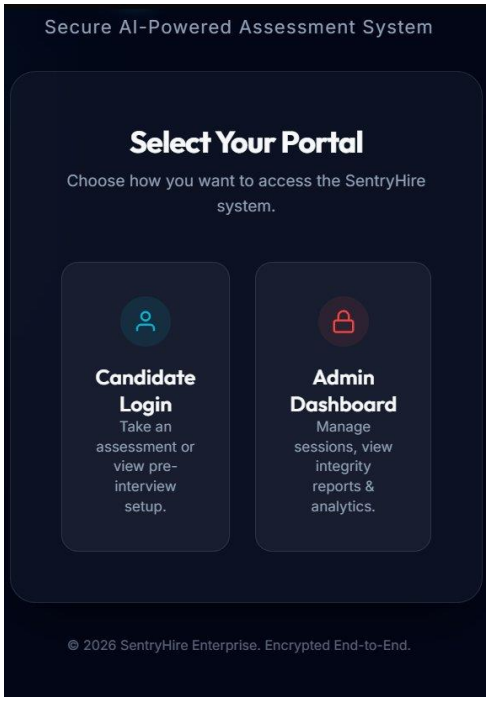

Fig. 1. Main Portal Selection Screen (index.html)

II. RELATED WORK

The most closely related work is that of Ramana Babu et al. [1], which proposed a post-interview fraud detection system using audio-visual synchronization. That system segments recorded interview videos into clips, extracts text independently from lip movements using LipNet and from audio using Whisper, converts both streams into Word2Vec embeddings, and computes Cosine Similarity via a Siamese network. While effective for post-processing, it does not support real-time detection and is limited to a single detection modality.

Hussain et al. [2] proposed face recognition and eye tracking for examination cheating detection. Atoum et al. [3] developed a deep CNN-based student behavior analysis system for automated exam monitoring. Veerabadran et al. [4] demonstrated that COCO-SSD models reliably identify prohibited items in webcam feeds. Chetty and Wagner [5] showed that multi-speaker detection is a reliable proxy for external coaching.

The proposed system directly extends Ramana Babu et al. [1] by transitioning from post-interview analysis to real-time, multi-modal, browser-native detection integrating five additional detection modalities beyond audio-visual analysis.

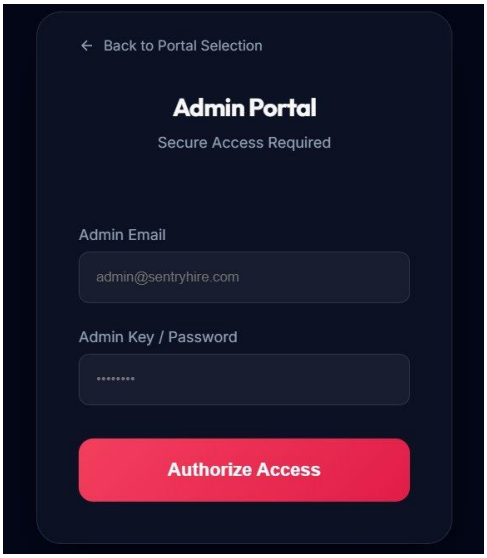

Fig. 2. Candidate Login Portal (candidate.html)

III. SYSTEM ARCHITECTURE AND METHODOLOGY

A. System Overview

The system is structured around two primary portals: a Candidate Portal through which the interviewee undertakes the monitored session, and an Admin Dashboard through which hiring administrators review session results. Firebase Realtime Database handles authentication, cross-device synchronization, and session record storage. All detection modules operate client-side within the candidate's browser, enabling low-latency inference without server-side processing.

The candidate workflow proceeds through five stages: (1) Firebase Authentication via email and password; (2) interview prerequisites acknowledgment and consent; (3) QR code-based mobile device synchronization; (4) live monitored session; and (5) session report generation with integrity score.

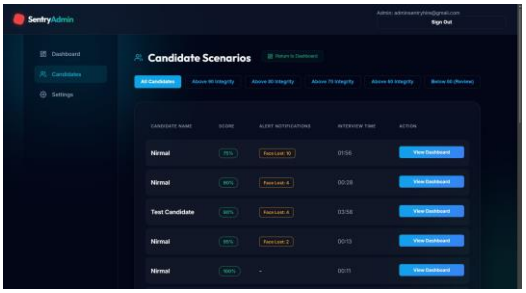

Fig. 3. Interview Prerequisites and Consent Screen (prerequisites.html)

B. Object and Face Detection Module

The object detection module employs TensorFlow.js with the COCO-SSD model to continuously analyze the candidate's webcam feed. The system detects and flags prohibited items including mobile phones, books, papers, tablets, and secondary laptop screens. Upon detection of any flagged object class, the AntiGravitySignalTracker (AGST) — a custom signal management class implemented in session_logic.js — fires a timed alert after a configurable activation duration, preventing false positives from transient detections.

MediaPipe Face Detection simultaneously monitors for face presence. If no face is detected for more than 2,000 milliseconds, a 'Face Not Detected' alert is triggered. The system also detects multiple faces in frame, flagging the presence of a potential unauthorized third party.

C. Eye Gaze and Head Pose Estimation

Using MediaPipe Face Mesh, the system tracks 468 facial landmarks per frame including iris landmarks (indices 469–477) to estimate gaze direction. The AGST monitors five sustained gaze conditions from real landmark coordinates: gaze_left and gaze_right (iris X offset from nose > 0.07, 3,000 ms activation, 20,000 ms cooldown), looking_down (iris Y

position relative to nose, threshold 0.08, 2,500 ms activation, 15,000 ms cooldown), head_turned (nose X offset from centre > 0.15, 2,000 ms activation, 15,000 ms cooldown), and eyes_closed (Eye Aspect Ratio < 0.02, 3,000 ms activation, 20,000 ms cooldown). A repeated gaze pattern detector logs deviations over a 60-second rolling window; three or more deviations in the same direction fire a combined high-confidence violation.

D. Multi-Voice and Whisper Detection

Audio analysis operates on two levels. First, the Web Audio API AnalyserNode (FFT size 2048, smoothing 0.8) monitors speech frequency bins 3–15 (approximately 70–350 Hz) for multiple voice peaks. If more than one distinct frequency peak (voicePeaks > 1) is detected across 3 consecutive frames with a 5,000 ms cooldown, a "Multiple Voices" alert is triggered. Second, the AIDetectionSystem class (ai_detection.js) initialises a Web Audio API context at 16,000 Hz and checks audio energy at 500 ms intervals. If energy falls below 30% of the adaptive baseline, a whisper signal is activated. For advanced speaker verification, a SpeechBrain ONNX model (spkrec-ecapa-voxceleb) is loaded via ONNX Runtime Web; a cosine distance threshold of 0.65 between sequential speaker embeddings identifies a second distinct speaker.

E. QR Code-Based Mobile Device Tracking

At session commencement, the candidate scans a unique QR code generated by the qrcodejs library, linking their mobile device to the session via Firebase. The mobile interface accesses the device's accelerometer via the DeviceMotion API. A smart calibration phase averages 25 readings to establish the resting baseline. Any subsequent total displacement exceeding 1.2 m/s² triggers a MOVEMENT_ALERT event written to Firebase, received in real time by the desktop session listener.

Fig. 4. QR Code Device Synchronization Screen (device_sync.html)

F. Tab Switch and Focus Detection

The Page Visibility API's visibilitychange event and the window blur event are monitored throughout the session. Any navigation away from the interview tab triggers an immediate 'Tab Switch Detected' alert. A 1,000 ms delay is applied to the blur handler to prevent false positives from accidental browser UI interactions.

G. Integrity Scoring and Alert System

Each session begins at an integrity score of 100. The triggerAlert() function applies a deduction of 2.5 points per alert, subject to a 2,500 ms spam cooldown between alerts of the same type. All alerts are timestamped and logged in the Violation Log panel. The AGST implements per-signal cooldown management — gaze deviation alerts carry a 20,000 ms cooldown, face loss alerts a 15,000 ms cooldown — ensuring sustained violations are flagged proportionally. At session end, the report is pushed to Firebase under completed_sessions and stored locally for report.html display.

IV. IMPLEMENTATION

The system is implemented using vanilla JavaScript, HTML5, and CSS3, deployed as a static web application on Netlify. Key libraries and technologies:

- MediaPipe Face Mesh and Face Detection — facial landmark tracking, iris detection, gaze estimation
- TensorFlow.js with COCO-SSD — real-time object detection in webcam feed
- Web Audio API — audio energy monitoring, whisper detection, multi-voice peak analysis
- ONNX Runtime Web + SpeechBrain — speaker embedding inference for multi-voice detection
- Transformers.js — Xenova/ast-finetuned-audioset AI voice classification
- Firebase Realtime Database — session storage, mobile sync, admin dashboard
- qrcodejs — QR code generation for mobile device pairing
- Page Visibility API — tab switch detection
- DeviceMotion API — accelerometer-based mobile movement tracking

The AntiGravitySignalTracker class centralizes all signal management, implementing per-signal activation timers, cooldowns, combined signal detection, and a text-to-speech warning system using Kokoro TTS via Transformers.js with a native SpeechSynthesis fallback.

V. RESULTS AND DISCUSSION

Preliminary testing was conducted to evaluate the responsiveness and reliability of each detection module under standard indoor conditions.

The object detection module consistently identified prohibited items including mobile phones and secondary screens. As shown in Fig. 5 and Fig. 6, when a candidate held a mobile phone in front of the camera, the AI Guard panel immediately flagged 'Threat Detected' and the Violation Log recorded timestamped 'Restricted Device: Mobile phone detected in frame' alerts, reducing the integrity score from 100% to 57–62% over the test session.

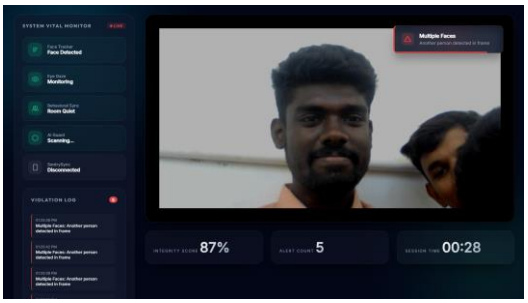

Fig. 5. Live Session – Mobile Phone Detected (Integrity Score: 57%, 17 Alerts)

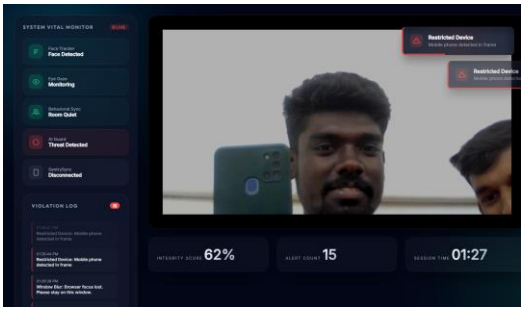

Fig. 6. Live Session – Mobile Phone + Multiple Faces Detected (Integrity Score: 62%, 15 Alerts)

Fig. 7 demonstrates multiple face detection. When a second person entered the camera frame, the system immediately displayed a 'Multiple Faces: Another person detected in frame' alert with an integrity score of 87% and 5 alerts logged within 28 seconds.

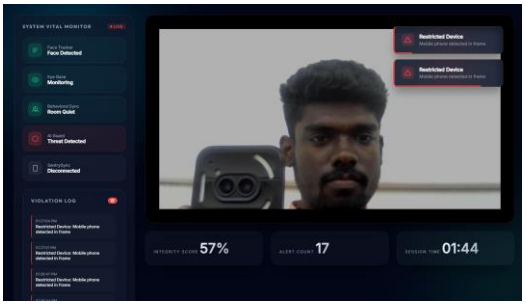

Fig. 7. Live Session – Multiple Faces Detected (Integrity Score: 87%, 5 Alerts)

Tab switch detection operated with 100% recall in all controlled tests. The QR-based mobile tracking system successfully delivered movement alerts to the desktop session with sub-500 ms Firebase latency across Android and iOS devices.

The admin dashboard (Fig. 8) displayed real-time system metrics including active session count, average integrity score (78%), total violations flagged (180), and a 7-day integrity score analytics chart. The Candidate Scenarios panel (Fig. 9) showed all historical sessions filterable by integrity score bands (Above 90, Above 80, Above 70, Above 60, Below 60 – Review), with per-candidate alert notifications, interview duration, and score.

Fig. 8. Admin Dashboard – System Overview (19 Active Sessions, 78% Avg Integrity)

Fig. 9. Admin Dashboard – Candidate Scenarios with Integrity Score Filtering

Fig. 10 shows the post-session SentryReport for candidate 'Nirmal' with a 97% integrity score, classified as

'High Integrity' with the message: 'The candidate maintained exceptional focus and followed all protocols perfectly.'

Fig. 10. Session Report – 97% Integrity Score, High Integrity Verdict

VI. LIMITATIONS AND FUTURE WORK

The current implementation has several acknowledged limitations. The eye gaze estimation module does not yet deliver directional precision sufficient for fine-grained fraud flagging under varying lighting conditions. The system currently represents the candidate-facing side of the platform; real-time integration with the interviewer's view remains a critical next step.

Future directions include: (1) integration as a browser extension for Google Meet and Zoom; (2) calibration-based gaze mapping for improved precision; (3) development of a real-time interviewer alert panel; (4) ATS integration; and (5) comprehensive usability studies.

VII. CONCLUSION

This paper presented an Automated AI System for Interview Fraud Analysis and Alerts, a browser-native web application integrating seven real-time detection modalities — object detection, face monitoring, eye gaze estimation, multi-voice detection, QR-based mobile tracking, and tab switch detection — into a unified integrity monitoring platform for online interviews.

The system's browser-native architecture, powered by MediaPipe, TensorFlow.js, and the Web Audio API, enables real-time client-side inference. The dynamic integrity scoring mechanism provides a quantitative summary of candidate behavior, while the Firebase-backed admin dashboard enables post-session assessment at scale. The system directly extends prior work on audio-visual synchronization-based fraud detection, representing a meaningful step toward secure and fair online interview environments.

VIII. ACKNOWLEDGMENT

The authors would like to express sincere gratitude to Mrs. K. Menaka, Assistant Professor, Department of Computer Science and Engineering, Sree Sowdambika College of Engineering, for her valuable guidance and continuous support throughout this project.

Project Demo: https://heartfelt-torrone-6e9673.netlify.app

REFERENCES

[1] B. Ramana Babu, Y. Sireesha, R. Suresh, V. Sai Srinivas, V. Hemanth, and S. Sumendra Kumar, "Fraud Detection in Online Interview Using Audio Visual Synchronization," IJSREM, vol. 9, no. 6, June 2025. DOI: 10.55041/IJSREM50894.
[2] M. Hussain et al., "Automated Cheating Detection in Online Exams Using Facial Recognition and Eye Tracking," Int. J. Advanced Computer Science and Applications, vol. 10, no. 6, 2019.
[3] Y. Atoum et al., "Automated online exam proctoring," IEEE Trans. Multimedia, vol. 19, no. 7, pp. 1609–1624, 2017.
[4] V. Veerabadran et al., "Object Detection for Online Proctoring Using COCO-SSD," Proc. Int. Conf. Intelligent Computing, pp. 112–119, 2020.
[5] G. Chetty and M. Wagner, "Multi-modal speaker verification using voice and lip movement features," INTERSPEECH 2008, pp. 1939–1942.
[6] C. Lugaresi et al., "MediaPipe: A Framework for Building Perception Pipelines," arXiv:1906.08172, 2019.
[7] D. Smilkov et al., "TensorFlow.js: Machine Learning for the Web and Beyond," arXiv:1901.05350, 2019.