Cogni-Hire:AI-Interviewer and Insight Generator

Anugya, Bhuvan C, Juhitha Radha, Varla Aneesh
UG Students, Indian Institute of Science
{anugyaa, bhuvanc, juhitharadha, varlaaneesh}@iisc.ac.in

Abstract

Modern hiring is an inference problem performed under noise, bias, and evaluator inconsistency. We build an end-to-end AI interview system that unifies resume parsing, role-conditioned question generation, speech and vision analysis, and a dual-manifold technical evaluator combining geometric similarity with LLM adjudication. The entire pipeline implemented through a React–FastAPI architecture produces stable, reproducible, and insight-rich candidate profiles. This report outlines the system's design and the principles that make it a rigorous alternative to human-only assessment. Our code is available at [github repo]

Architectural Decomposition

The platform decomposes naturally into five interacting subsystems:

1. **Audio Analysis:** speech clarity, pacing, prosody, and communication features.
2. **Question Generation**: Role and Resume Conditioned Question Synthesis.
3. **Malpractice Detection** using computer vision and behavioral signals.
4. **Dual-Manifold Technical Evaluation** blending geometric similarity with LLM adjudication.
5. **Structured Report Generation**: consolidates all computed metrics and insights into a structured final report.
6. **Orchestration Layer:** React frontend, employer dashboard, and a FastAPI backend.

Taken together, these modules form an end-to-end pipeline from raw applicant data to a structured evaluation report.

Audio Analysis Pipeline

Dual-Path Processing

The candidate's audio is processed through two parallel channels:

- **Content Path (ASR)**: Whisper provides high-fidelity speech-to-text.
- **Acoustic Path**: The raw waveform is analyzed for fluency, hesitations, and prosody.

Acoustic Feature Extraction

Fluency is measured using words-per-minute (WPM), discounting silent segments. **Hesitation analysis** detects fillers and long unnatural pauses. **Prosody** uses pitch variance to infer engagement and vocal confidence.

.

Communication Score

The final communication score is computed from:

- Fluency Score (Gaussian penalty model)
- Clarity Score (pause and filler penalties)
- Engagement Score (pitch-variance normalization)

Question Generation

Role-Based Questions

Questions were curated from the AI Recruitment Pipeline dataset, cleaned, normalized, and stored in structured JSON form. During an interview, 5–7 questions are sampled per role. The system is compatible with FAISS and Sentence-BERT for retrieval-based question similarity. Each question includes a canonical answer for evaluation.

Resume-Based Questions

Resumes are parsed using PyPDF2 and analyzed by an LLM, which extracts skills and projects in JSON mode. The LLM generates personalized questions, followed by up to two adaptive follow-up questions per answer. This creates a tailored interview aligned with the candidate's background.

Malpractice Detection

Remote interviews introduce opportunities for misconduct. Our system mitigates this using four detection channels:

1. **Multiple Face Detection**: TinyFace flags more than one face across frames.
2. **Gaze Tracking**: OpenFace estimates gaze vectors and flags prolonged off-screen behaviour.
3. **Tab/Window Switching**: Browser APIs detect focus-loss and tab switching.
4. **Response Delay**: Delays exceeding 10 seconds between question completion and speech onset trigger suspicion.

An aggregated malpractice score weights incidents without automatic disqualification.

Evaluation

Dual–Manifold Representation. Let $\mathcal{X}$ denote the space of free-form answers. We embed each answer $x \in \mathcal{X}$ into two orthogonal manifolds:

$$\phi_K : \mathcal{X} \to \mathbb{R}^{128}, \qquad \phi_R : \mathcal{X} \to \mathbb{R}^{32},$$

where ϕ_K captures *knowledge correctness* (the "static content") and ϕ_R captures *reasoning trajectory* (the "dynamic path"). Given candidate answer a and canonical solution c, we define:

$$d_K = 1 - \cos(\phi_K(a), \phi_K(c)), \qquad d_R = 1 - \cos(\phi_R(a), \phi_R(c)).$$

The per-question score is the structural blend

$$S_{\text{tech}} = \alpha(1 - d_K) + (1 - \alpha)(1 - d_R), \qquad \alpha = 0.7.$$

This mirrors Tao's philosophy of disentangling structure from randomness: the manifold ϕ_K measures proximity of ideas, while ϕ_R measures fidelity of reasoning flow. [paper]. Please refer [A4]

Strict LLM Adjudication. To recover subtleties inaccessible to pure geometry, we condition a state-of-the-art LLM with a highly constrained persona: refer to [A4.5]

The model produces four sub-scores Conceptual Correctness, Precision Grounding, Clarity, and Creativity and a one- line synthesis describing *trajectory*, *strengths*, *risks*, and a *recommendation*. The combined per-question score is:

$$S^*_{\text{tech}} = \tfrac{1}{2}\big(S_{\text{tech}} + \overline{S}_{\text{LLM}}\big).$$

CV Evaluation Manifold. A parallel rubric evaluates CVs across five analytically meaningful axes:

$$\text{Relevance},\ \text{Impact},\ \text{Trajectory Coherence},\ \text{Technical Specificity},\ \text{Career Momentum}.$$

Each is scored in $[0, 10]$ under the same strict persona & The quantity is characterized by its central tendency. The global competency index is:

$$\mathrm{CCI} = 0.75\, S^{*}_{\text{tech}} + 0.25\, \overline{S}_{\text{cv}}.$$

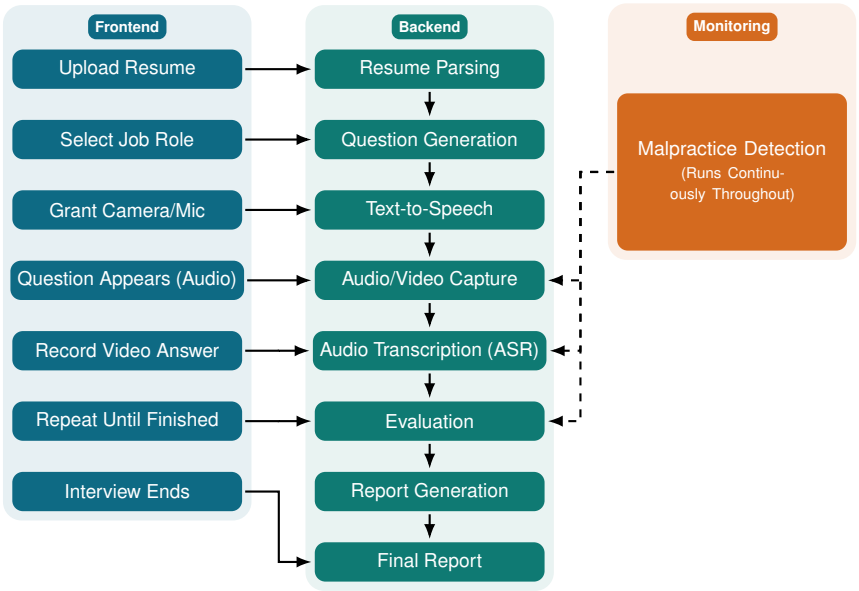
Thus the interview governs the dominant term, while the CV acts as a prior.

Report Generation

We generate a 2-page Latex report which consolidates all numerical outputs per-question scores, sub-component bars, and the five-axis radar profile into a single evaluative surface that exposes both stability and fracture points in the candidate's performance. Each graphic is paired with a distilled narrative analysis that interprets the geometry.

System Integration & Frontend

The complete pipeline consists of:



The frontend is implemented in React + Vite, with Axios-based communication to FastAPI services.Main Dadhboard leads to 2 Dashboards. Candidate dashboard allows users to upload their resume , select role and take the interview Employer dashboards provide session history, scores, flags, and PDF reports.

Conclusion

This system provides a scalable and unbiased alternative to manual interviews, offering reproducible assessments, richer behavioural signals, and consistent evaluation. It minimizes human bias, reduces cost, and introduces a diagnostic, structured understanding of candidate performance. Future work includes microservice scaling, expanded role libraries, and richer behavioural modelling.

Acknowledgements

We also thank Professor **Deepak Subramani** for his valuable inputs and for creating a well-structured course that enabled this work.

Individual Contributions

Radha Vegesna: Question Generation, Malpractice, Integration, Frontend, UI
Anugya Anantapur: Question Generation, Malpractice, UI/UX
Aneesh Varla: Integration, Input Processing
Bhuvan C: Evaluation, Report Generation

Appendix

A1. Retrieval-Enabled Architecture (FAISS + Sentence-BERT)

To support adaptive and semantic question retrieval:

- All questions are embedded using Sentence-BERT (768-dimensional vectors).
- FAISS (IndexFlatIP) is used for fast similarity search.

Similarity is computed using cosine similarity:

$$\text{similarity}(q_1, q_2) = \cos\left(\text{SBERT}(q_1), \text{SBERT}(q_2)\right)$$

This enables scalable and domain-independent retrieval of related questions.

A2. Malpractice Threshold Logic

This section outlines the detection thresholds and scoring rules used to identify suspicious behaviour during remote interviews.

A3.1 Multiple Face Detection

Using TinyFace, the system counts faces per frame.

Threshold Logic:

- Face count ≥ 2
- Occurring in at least 3 frames
- Within a 10-second sliding window

Formally:

$$\sum_{t=1}^{10s} \mathbb{1}(\text{faces}_t \geq 2) \geq 3 \;\Rightarrow\; \text{multi-face flag}$$

A3.2 Gaze Tracking Thresholds

Gaze direction is estimated using eye landmarks and head pose.

Thresholds:

- Continuous off-screen gaze > 5 seconds $\Rightarrow$ warning
- Off-screen proportion $> 30\%$ for any question $\Rightarrow$ flag

$$P_{\text{offscreen}} = \frac{\text{frames}_{\text{off}}}{\text{frames}_{\text{total}}} > 0.30 \Rightarrow \text{flag}$$

A3.3 Window & Tab Switching Detection

Browser events: `visibilitychange`, `blur`, `pagehide`, and `document.hasFocus()` are used.

Threshold Rules:

- Focus-loss < 2 seconds $\Rightarrow$ ignored (transient)
- Focus-loss ≥ 2 seconds $\Rightarrow$ suspicious event
- More than 2 focus-loss incidents $\Rightarrow$ escalation

$$N_{\text{tab-switch}} > 2 \;\Rightarrow\; \text{malpractice escalation}$$

A3.4 Response Delay Detection

Response delay is computed as:

$$d = t_{\text{speech-start}} - t_{\text{question-end}}$$

This will account for gap between question and start recording button **Threshold:**

- Delay > 10 seconds $\Rightarrow$ suspicious delay flag

A3.5 Weighted Malpractice Scoring Model

Each incident type contributes to the malpractice score:

Incident Type	**Weight**
Multiple faces detected	3
Off-screen gaze $> 30\%$	3
Tab/Window switching	3
Response delay > 10s	3
Identity mismatch	3
Repetitions of same incident	+1 per repeat

Final score is the weighted sum of all incidents.

A4. Evaluation Formulation & report generation

This appendix records the mathematical structure underlying our evaluation engine.

A4.1. Local Linear Approximation of Embedding Geometry

Let $\mathcal{X}$ denote the space of free–form responses. Modern encoder models (SentenceTransformers: `all-mpnet-base-v2`, `MiniLM-L6-v2`) produce embeddings

$$\phi_K(x) \in \mathbb{R}^{128}, \qquad \phi_R(x) \in \mathbb{R}^{32}.$$

Although the ambient maps are learned, the embedding spaces admit a *local linear approximation*:

$$\mathcal{M}_K \approx T_c\mathcal{M}_K, \qquad \mathcal{M}_R \approx T_c\mathcal{M}_R,$$

where T_c denotes the tangent space at the canonical solution c. Under this approximation, angular displacement provides a scale–free proxy for the normalized geodesic length:

$$d_K = 1 - \cos(\phi_K(a), \phi_K(c)), \qquad d_R = 1 - \cos(\phi_R(a), \phi_R(c)).$$

This treats the manifolds as locally spherical, allowing cosine distance to function as the first–order surrogate for intrinsic separation.

A4.2. Product–Manifold Structure

The evaluation space is the Cartesian product

$$\mathcal{M} = \mathcal{M}_K \times \mathcal{M}_R,$$

where

- $\mathcal{M}_K$ encodes static knowledge alignment,
- $\mathcal{M}_R$ encodes dynamic reasoning trajectory.

Each sub-manifold contributes orthogonal information. An element of the product space is written

$$z = (u, v), \qquad u \in \mathcal{M}_K,\ v \in \mathcal{M}_R.$$

A4.3. Weighted Product Metric

The scoring metric corresponds to a weighted product metric on $\mathcal{M}$:

$$\langle (u_1, v_1), (u_2, v_2) \rangle_\alpha = \alpha \langle u_1, u_2 \rangle + (1 - \alpha) \langle v_1, v_2 \rangle,$$

with $\alpha = 0.7$ chosen to privilege correctness over trajectory.

Under this metric, the induced distance reduces (via normalization) to the blended score:

$$S_{\text{tech}} = \alpha(1 - d_K) + (1 - \alpha)(1 - d_R).$$

Thus the technical score is nothing more than the similarity on a weighted product manifold, computed using local angular coordinates.

A4.4. Angular Metrics

In locally spherical coordinates, cosine similarity depends only on the angular displacement, and is therefore invariant under "destructive steps" extraneous digressions or linguistic inflation that modify the *path length* of the linguistic sequence but not its *direction* in embedding space.

Formally, for embeddings normalized to unit length,

$$\phi(x) = \frac{\psi(x)}{\|\psi(x)\|},$$

the score depends solely on the angle

$$\theta = \arccos\langle \phi(a), \phi(c) \rangle,$$

so perturbations orthogonal to the canonical direction do not artificially inflate similarity. This provides structural robustness: only idea–alignment and trajectory fidelity contribute.

A4.5. Integration with the Strict LLM Adjudicator

While the manifold score detects structural proximity, it cannot recover higher order qualities (clarity, precision, creativity). We therefore condition a large language model (Gemini 1.5 pro) with an IMO judge persona

> *"Evaluate as a panel of IMO judges. Scores in* $[8, 9]$ *require genuine substance & rigorous development of the core ideas.; scores* > 9 *demand rare conceptual elegance.; scores in* $[4, 7]$ *captures clear flow of thought and a logical approach, even if the final result or execution is flawed or incomplete.; scores* < 4 *Indicates little to no relevant progress, or a completely disjointed/confused approach to the problem."*

to produce the vector

$$S_{\text{LLM}} = \big(S_{\text{conceptual}},\ S_{\text{precision}},\ S_{\text{clarity}},\ S_{\text{creativity}}\big),$$

$$\overline{S}_{\mathrm{LLM}} = \frac{1}{5}\sum_{i=1}^{5} S_{\mathrm{LLM}}^{(i)},$$

where $\overline{S}_{\mathrm{LLM}}$ denotes the per-candidate LLM adjudication score, obtained as the simple average of the five sub-scores returned by the model.
and it also generates a one-line synthesis describing:

$$\text{trajectory}, \quad \text{strengths}, \quad \text{risks}, \quad \text{recommendation}.$$

The combined per-question score is:

$$S^{*}_{\text{tech}} = \tfrac{1}{2}\big(S_{\text{tech}} + \overline{S}_{\mathrm{LLM}}\big).$$

A4.6. Software Stack

- **Embedding models:** SentenceTransformers (`all-mpnet-base-v2`, `MiniLM-L6-v2`).
- **LLM adjudicator:** Google Gemini 1.5 Pro (prompts).
- **Numerical stack:** NumPy / SciPy for cosine geometry.
- **Visualization:** Matplotlib (radar charts), Seaborn.
- **Report synthesis:** LaTeX (`xelatex`) with custom templates.
- **Backend:** FastAPI.

A5. Frontend & Backend Architecture

The system uses a simple client-server design: the React frontend manages the interface and recordings, while the Python backend handles question generation, transcription, evaluation, and report creation. Both communicate through clean API calls, with malpractice detection running in the background throughout the interview.

A6. API Endpoints

The frontend interacts with backend components through a sequence of API operations:

- Upload Resume API
- Start Interview API
- Malpractice Log API
- End Interview API
- Get Roles API
- Process Answer API
- List Sessions API
- Get Session API
- List Reports API
- Get Report API
- Get Report PDF API

References

[1] OpenAI Whisper, Automatic Speech Recognition Model.

[2] FAISS: Facebook AI Similarity Search.

[3] Reimers, N., Gurevych, I. "Sentence-BERT: Sentence Embeddings using Siamese BERT Networks" (2019).

[4] OpenFace Facial Behaviour Analysis Toolkit.

[5] PyPDF2 Documentation.

[6] FastAPI Documentation.

[7] React & Vite Official Documentation.

[8] Google Gemini API Reference.