AI-Based System for Interview Automation and Candidate Insight Generation - 2 month Project Proposal

TeamID : [16]

Project Overview

Development of an AI-powered Interview System designed to conduct and evaluate role-specific interviews across technical domains. The system interviews candidates, evaluates their responses against a structured rubric, and adapts its questioning dynamically through context-aware follow-ups. Leveraging large language models (LLMs) for dialogue management and semantic analysis, the AI ensures that questions are both role-specific and adaptive to candidate performance.

At the conclusion of each session, the system generates a comprehensive candidate report summarizing strengths, areas for improvement, and rubric-based scoring. This enables recruiters to achieve scalable, consistent, and unbiased evaluations while reducing manual effort in the hiring process.

Business Value Proposition

For Companies / Recruiters:

- **Scalability:** Screen thousands of candidates simultaneously, enabling high-volume hiring drives with ease.
- **Cost Reduction:** Minimize the need for large HR teams in the early stages of recruitment, reducing operational costs by up to 40%.
- **Fairness & Consistency:** Reduce human bias, ensuring standardized evaluation across applicants with rubric alignment to expert interviewers.
- **Data-Driven Insights:** Get detailed analytics about candidate performance.

For Candidates:

- **Actionable Feedback:** Unlike traditional interviews, Receive structured evaluation reports that highlight strengths, weaknesses and areas for improvement.
- **Fair Opportunity:** Reduced chances of bias related to gender, accent, or personal preferences.
- **Skill Development:** Leverage actionable insights to prepare more effectively and build stronger competencies for future opportunities.

Tech Architecture

Overview

This architecture focuses exclusively on the AI pipeline — no frontend or full product infra. The model stack performs: speech-to-text, adaptive question selection and follow-ups, rubric-based answer evaluation, simple malpractice detection from audio signals, paralinguistic (confidence) analysis, and automated report generation.

Primary Layers

- **Input Processing:** Audio ingestion → ASR → cleaned transcript; extract audio features (VAD, pauses, pitch).
- **Question Retrieval & Management:** Role-tagged question dataset + embedding index for retrieval.
- **Adaptive Question Generator:** LLM-based module to produce next question or follow-up conditioned on context.
- **Answer Evaluator:** Embedding-based semantic matching + fine-tuned rubric classifier to produce per-question scores.
- **Malpractice Detector (AI-only):** Diarization / multiple-voice detection, long silence/sudden background-change detectors.
- **Emotion/Confidence Estimator:** Feature-based classifier (hesitation, speaking rate, pitch variance) that yields confidence indicators.
- **Report Generator:** LLM summarizer + structured JSON output containing Rubric scores, Notes, Highlights and Flags.

Detailed Pipeline

1. Audio → Transcript

- Uses **Whisper / WhisperX** (or equivalent) for ASR; output timestamped transcript segments.
- Post-process: punctuation restoration, filler removal (uh/um), simple normalization.
- Metrics to track: Word Error Rate (WER).

2. Paralinguistic Feature Extraction

- Extract features per utterance: pause durations, speech rate (words/min), pitch mean/variance, energy, stutter counts (repetition patterns).
- Optional: compute short-term MFCC / OpenSMILE feature vectors for emotion models.

3. Question Retrieval and Context

- Store question bank (role, competency, difficulty, canonical answers, rubric examples) as plain text + metadata.
- Precompute embeddings (Sentence-BERT) for questions; store in FAISS / on-disk index for retrieval.
- Retrieval logic: role + previous answers + desired competency → top-N candidate questions.

4. Adaptive Follow-up Generation (Prompt Pipeline)

- Uses an LLM (GPT-5, LLaMA or equivalent) with a staged prompt pipeline for better control and transparency.
- **Stage 1: Understanding Candidate Answer**

  > ”Summarize the candidate’s response concisely, highlighting the main reasoning and key points.”

- **Stage 2: Gap Analysis Against Rubric**

  > ”Compare the candidate’s answer with the rubric for role {role}. Identify missing concepts, incomplete explanations, or unclear reasoning.”

- **Stage 3: Context-Aware Follow-up Question**

  > ”Based on the identified gaps, generate one focused follow-up question tailored to role, probing deeper into the missing areas.”

- Each stage ensures modularity, transparency, and easier debugging compared to a single prompt.
- Length is constrained; no biased or leading phrasing is permitted.

5. Answer Evaluation (Rubric)

- Compute embedding of candidate answer (Sentence-BERT / OpenAI embeddings).
- Semantic similarity against canonical answers / rubric exemplars (cosine similarity).
- Feed features (embedding, length, paralinguistic signals) to a fine-tuned classifier/regressor that maps to rubric scores (e.g., 1–10 per competency).
- Calibrate scores with dataset .

6. Malpractice Detection (AI-only)

- Use diarization (pyannote) to detect extra voices; if more than one active speaker detected during candidate response → flag.
- Detect long, repeated silent segments or abrupt background changes (possible external help) and flag with timestamps.
- These flags are soft: surface them in the report for human review.

7. Emotion / Confidence Estimation

- Train a small classifier on extracted features to predict confidence/hestitation/notable stress indicators.
- Output one or two interpretable signals (e.g., *confidence: high/medium/low*, *stutter-index*).

8. Report Generation

- Compile per-question rubric scores, overall weighted score, confidence metrics, and malpractice flags.
- Use LLM summarizer to generate a concise narrative: strengths, improvement points, and recommended actions.
- Export formats: structured JSON (primary) and an optional PDF/HTML summary.

Datasets [Open Source]

Modality	**Dataset**	**Purpose / Usage**
Text	AI Recruitment Pipeline Dataset (Q+A+Decisions)	Core training data: evaluates candidate answers against recruiter decisions.
	Kaggle Software Engineering Questions	Technical interview questions for role-specific evaluation.
Speech	RAVDESS	Emotional speech dataset for confidence and emotion detection.
	IEMOCAP	Dialogues with labeled emotions, closer to natural interviews.
	Mozilla Common Voice	Multilingual dataset for fluency and accent robustness.
Video (Optional)	AffectNet	1M+ facial images labeled with emotions for expression analysis.
	FER2013	Facial emotion recognition dataset (smaller, benchmark).
Security/Integrity	SiW (Spoof in the Wild)	Detects face spoofing (photo, replay, mask attacks).
	ASVspoof	Detects replayed or AI-generated voices.

Table 1: Summary of datasets required for different modules of the AI Interview Assessor.

Timeline

Phase 1: Foundation & Core Pipeline (Weeks 1-3)

Weeks	Key Tasks
1-2	• Finalize the detailed AI pipeline architecture and model choices. • Set up development environment: ASR model access (Whisper), LLM API keys, and vector DB (FAISS). • Source and preprocess the core question bank and the AI Recruitment Q&A dataset.
3	• Implement the ‘Audio - Transcript‘ module using Whisper/WhisperX. • Build the ‘Question Retrieval‘ engine with Sentence-BERT embeddings and a FAISS index. • Develop a baseline ‘Answer Evaluator‘ using semantic similarity.
	Environment setup and processed datasets complete and Baseline Pipeline Prototype done

Phase 2: Adaptive Intelligence & Paralinguistic Analysis (Weeks 4-6)

Weeks	Key Tasks
4-5	• Implement the three-stage ‘Adaptive Follow-up Generation‘ LLM prompt pipeline. • Develop the ‘Paralinguistic Feature Extraction‘ module (pauses, speech rate, pitch). • Train the initial ‘Emotion/Confidence Estimator‘ classifier on extracted audio features.
6	• Fine-tune the ‘Answer Evaluator‘ classifier using text embeddings and paralinguistic features. • Calibrate rubric scoring against the human-graded dataset to meet alignment targets.
	Intelligent Multi-Modal Evaluation with Adaptive Scoring and Confidence Analysis is complete.

Phase 3: Integrity, Reporting & Finalization (Weeks 7-9)

Weeks	Key Tasks
7-8	- Implement the ‘Malpractice Detector‘ using speaker diarization (pyannote) for voice detection. - Develop the final ‘Report Generator‘ using an LLM to create narrative summaries and structured JSON output. - Conduct end-to-end pipeline integration testing.
9	- Evaluate the full system against defined targets (WER, Rubric Alignment, Latency). - Finalize bug fixes and performance tuning. - Prepare final project documentation and presentation.
	Final AI Pipeline (V1.0) with comprehensive evaluation results and documentation and All AI modules feature-complete and integrated; reporting is functional

Evaluation Targets

- ASR WER: $\leq 10\%$ on interview-style audio (after basic tuning).
- Rubric alignment: $\geq 80\%$ agreement with professor labels (primary quantitative goal).
- Emotion/confidence accuracy: $\approx 70\% - 80\%$ on curated test set.
- Malpractice detection recall: $\geq 80\%$ on synthetic/evaluation scenarios (human-in-loop verification required).
- Latency: single-question evaluation $\approx$ 1–3 seconds (model-only pipeline).

Deliverables

- **Working AI Interview System:** End-to-end system capable of conducting interviews in speech, and optionally video, capturing candidate responses and storing structured logs.
- **Adaptive Questioning Capability:** Context-aware question selection and follow-up generation tailored to role and previous responses.
- **Candidate Evaluation Reports:** Automated generation of per-question scores, overall performance summaries, confidence indicators, and flagged integrity issues.

- **Malpractice Detection Outputs:** Detection and flagging of potential cheating or anomalies, including voice/spoofing irregularities and abnormal audio patterns.
- **Preprocessed Multi-Modal Dataset:** Feature-extracted text, audio, and optional video datasets for model training and evaluation.