DOCUMENTATION SUMMARY REPORT

PROJECT NAME: Interview ACE

EVENT: UpSkill India Challenge, Techfest IIT Bombay 2025.

1. Problem Statement

Technical proficiency alone does not guarantee interview success. A significant "code-to-communication" gap exists where qualified engineers fail because they cannot articulate complex concepts under synchronous pressure. Current preparation platforms are text-based and asynchronous, failing to simulate the cognitive load of real-time, spoken technical interrogation. Candidates lack a safe environment to practice verbal delivery, trade-off analysis, and structured thinking against an adaptive adversary. The industry requires a multimodal, voice-first solution that moves beyond syntax verification to evaluate a candidate's ability to communicate, reason, and adapt in a high-stake, "Bar Raiser" interview environment.

2. Approach and AI Components

Our platform bridges the gap between static coding tests and live interviews through a **Voice-First, Multimodal Feedback Loop**. Unlike text-based chatbots, our system processes Base64-encoded audio directly from the browser, forcing candidates to practice verbal articulation—the highest-bandwidth channel in real engineering interviews.

The "Bar Raiser" Persona: At the core is a custom orchestration module (llm.py) leveraging **Google Gemini 2.5**. We move beyond simple Q&A by injecting dynamic system prompts that instruct the model to adopt a strict "Senior Engineer" persona. This persona does not passively accept answers; it identifies vague responses and triggers adaptive probing questions to drill down into edge cases, simulating the stress of a real "Bar Raiser" round.

Structured Evaluation Engine: To eliminate subjective bias, we utilize **Deterministic Output Parsing** via Pydantic models. The evaluation microservice constrains the LLM to adhere to a strict JSON schema, ensuring feedback is delivered not as generic text, but as quantifiable metrics (0-100 scores) across three distinct axes: *Technical Accuracy*, *Communication Clarity*, and *Structural Depth*. This ensures consistent, data-backed performance tracking.

3. Technical Architecture Overview:

Frontend:

Built on **Next.js 14 (React)**, the client utilizes the App Router for superior rendering performance and static asset serving via **Netlify**. Its primary technical responsibility is the **efficient preparation of input data**. This is managed by a custom useAudioRecorder hook, which captures the user's audio input, performs client-side compression, and transforms the data into lightweight **Base64-encoded strings** before transmission, minimizing network load.

Backend:

The core business logic runs on **FastAPI (Python)**, deployed to **Render** with asynchronous **Uvicorn** workers. This choice provides high concurrency, which is essential for managing I/O-bound operations like long-running LLM calls without blocking the server. It handles three core services: Audio Ingestion, LLM Orchestration (using the llm.py module), and Deterministic Scoring. Data integrity is enforced rigorously using **Pydantic** models to validate the schema of all incoming requests and outgoing responses.

Database:

Firebase Firestore serves as the transactional database, crucial for maintaining **LLM Contextual State**. Session history is stored in a NoSQL document model (Sessions -> Q&A Pairs), allowing for fast retrieval of the entire **Context Window** required by the AI for adaptive follow-up questions. This real-time synchronization also forms the foundation for future live-feedback features.

4. Challenges and Mitigations:

Challenge 1: CORS & Mixed-Origin Policy

Mitigation: Explicitly whitelist trusted origins (Netlify URL, localhost) in FastAPI middleware.

Challenge 2: LLM Hallucination in Scoring

Mitigation: Implemented **pre-inference validation** middleware that checks for minimum transcript length before invoking Gemini. Empty or insufficient sessions automatically default to a score of zero, preventing the LLM from generating scores based on absent data.

5. Roadmap to Final Build:

Current Features:

1. Real-time voice-based interviews
2. Cloud Deployment
3. Custom Job Roles and Descriptions
4. Difficulty level (easy, medium, hard)
5. Strict, structured scoring using Gemini
6. Session tracking and storage
7. Detailed Feedback report with answer ratings
8. Areas for improvement
9. Auto generated improvement plan and suggested learning materials

Next Steps (To be done On-Site):

1. Resume Parsing (Upload CV (pdf) for more context)
2. Support for Video Interviews (using cv2 and new API endpoint to analyze video for confidence detection)
3. Multi Round Simulation (Includes the round number and tracks candidate scores persistently)

60-80% Prototype Completion Reached