Foundations of Artificial Intelligence

Real-Time AI Voice Interviewing Agent: Mock Interview System for Next-Gen Job Seekers

A PROJECT REPORT

Submitted by

2116230701159 – KUMARAN S

2116230701133 – KABILESH P

RAJALAKSHMI ENGINEERING COLLEGE

October, 2025

BONAFIDE CERTIFICATE

Certified that this project report "Real-Time AI Voice Interviewing Agent: Mock Interview System for Next-Gen Job Seekers" is the bonafide work of "Kumaran S(230701159) & Kabilesh P(230701133)" who carried out the project work under my supervision.

SIGNATURE OF THE FACULTY INCHARGE

Submitted for the Practical Examination held on _________________

SIGNATURE OF THE INTERNAL EXAMINER

TABLE OF CONTENTS

ABSTRACT

Job interviews are critical, high-stakes events that significantly impact career trajectories. However, effective preparation remains a challenge for many aspirants due to the limitations of existing tools and the high cost of professional coaching. Traditional mock interview platforms often rely on text-based interactions or generic question sets, failing to simulate the dynamic, real-time nature of actual interviews and neglecting crucial verbal communication skills. This project, titled "Real-Time AI Voice Interviewing Agent," introduces an innovative solution to bridge these gaps by leveraging Artificial Intelligence (AI) to create a realistic, adaptive, and accessible mock interview environment.

The system provides an intelligent platform where users engage in voice-based mock interviews with an AI agent. This agent utilizes advanced Natural Language Processing (NLP), Speech-to-Text (STT), and Text-to-Speech (TTS) technologies to conduct natural-sounding conversations. A key innovation is the system's adaptive workflow; the AI interviewer dynamically adjusts questioning based on the candidate's responses, role requirements, industry focus, and desired skill level, mimicking the interactive nature of human-led interviews.

This project employs a modern technology stack including Next.js for a responsive frontend, Tailwind CSS for UI styling, Firebase for backend user management and data storage, and the Vapi SDK integrated with Large Language Models (LLMs) like Google's Gemini for sophisticated voice interaction and feedback generation. Following the interview, the system provides immediate, comprehensive feedback analyzing aspects such as speaking pace, fluency, clarity, tone, content relevance, and confidence markers. Performance analytics are tracked over time, allowing users to monitor their progress and identify specific areas for improvement.

By offering a voice-first, adaptive, and feedback-rich preparation experience, this AI-powered system aims to democratize interview coaching, making high-quality practice accessible to all job seekers. It addresses the shortcomings of current tools by providing realistic simulations and actionable insights, ultimately empowering candidates to build confidence and perform effectively in career-defining moments. This project demonstrates the transformative potential of conversational AI in revolutionizing professional skill development and interview preparation.

1. INTRODUCTION

Securing employment in today's competitive job market hinges significantly on a candidate's ability to perform well during interviews. Interviews serve as the primary gateway for employers to assess not only technical skills and qualifications but also crucial soft skills such as communication, critical thinking, and cultural fit. Despite the undeniable importance of interview proficiency, many job seekers, particularly recent graduates, career changers, and those facing financial constraints, struggle to adequately prepare. The process often involves navigating a landscape of preparation tools and resources that fall short of replicating the authentic pressures and dynamics of a real interview scenario.

The existing paradigm for interview preparation presents several significant challenges. Traditional methods, such as reviewing common questions online or practicing responses in front of a mirror, lack interactivity and objective feedback. While online platforms offering mock interviews have emerged, many are text-based, completely missing the vital element of verbal communication – a skill paramount in most professional roles. These platforms often present static, one-size-fits-all question banks that do not cater to specific job roles, industries, or the candidate's individual experience level. Furthermore, solutions that do offer more realistic simulations or human coaching are frequently positioned towards recruiters and hiring managers, or they come with prohibitive costs, often ranging from $50 to $150 per session. This affordability crisis creates a significant barrier, limiting access to high-quality preparation for a large segment of the job-seeking population. Consequently, many candidates enter critical interviews feeling underprepared, lacking confidence, and unable to effectively articulate their value, potentially missing out on career-defining opportunities.

The limitations inherent in current preparation methods highlight a distinct market gap: the need for an accessible, realistic, and personalized interview practice tool. The rise of Artificial Intelligence (AI), particularly in the domain of Natural Language Processing (NLP) and conversational AI, presents a unique opportunity to address these shortcomings. AI technologies can power systems capable of understanding spoken language, engaging in dynamic conversations, adapting questioning strategies in real-time, and providing nuanced feedback on both the content and delivery of a candidate's responses. Such a system could effectively simulate the spontaneity and interactive nature of a human interviewer, offering a practice environment that closely mirrors real-world conditions.

This project proposes the development of a "Real-Time AI Voice Interviewing Agent," a sophisticated mock interview system designed specifically for job seekers. Leveraging cutting-edge AI, including advanced voice recognition (STT), natural language generation (NLG), realistic speech synthesis (TTS), and intelligent workflow management, this platform aims to revolutionize interview preparation. The system will feature a voice-first interface, allowing users to practice speaking their answers naturally. The AI agent will employ adaptive questioning techniques, tailoring the interview flow based on the user's input, the target job role, and specified areas of focus (e.g., technical, behavioral). Crucially, immediately following the session, the AI will deliver detailed, actionable feedback on various performance aspects, including verbal fluency, tone, response structure, content relevance, and confidence indicators. By providing an affordable, on-demand, and highly realistic practice environment coupled with personalized coaching insights, this AI-powered system seeks to empower candidates, build their confidence, refine their communication skills, and ultimately enhance their chances of interview success in the competitive job market. This work explores the design, implementation, and potential impact of such an intelligent system, demonstrating how AI can effectively bridge the gap in modern interview preparation.

2. LITERATURE REVIEW

The development of AI-driven tools for human resources, recruitment, and professional skill development has seen significant growth in recent years. Specifically, the application of AI to interview preparation and assessment aims to overcome the limitations of traditional methods by offering scalability, consistency, and data-driven insights. This literature review explores existing research and systems related to automated interview assessment, conversational AI for coaching, and the specific technologies underpinning the proposed Real-Time AI Voice Interviewing Agent.

Early approaches to automating interview processes often focused on analyzing textual responses or non-verbal cues from video recordings. For instance, systems utilizing NLP techniques like Term Frequency-Inverse Document Frequency (TF-IDF) and early versions of Bidirectional Encoder Representations from Transformers (BERT) were explored for scoring written answers based on keyword matching and semantic similarity to ideal responses. However, these text-based methods inherently fail to capture the nuances of verbal communication, which is a critical component of most interviews. Research involving multimodal analysis, combining video (facial expressions, gestures) and audio (speech patterns, tone), sought to provide a more holistic assessment. Systems using Hidden Markov Models (HMMs) or Support Vector Machines (SVMs) combined with features like Mel-Frequency Cepstral Coefficients (MFCCs) for audio and Histogram of Oriented Gradients (HOG) for video demonstrated moderate success in predicting interviewer ratings. However, these systems often struggled with the complexity of real-time interaction, the need for large labeled datasets, and potential biases embedded in the training data.

The advent of more powerful language models, such as RoBERTa (A Robustly Optimized BERT Pretraining Approach), marked a significant advancement in automated content evaluation. Studies comparing RoBERTa with TF-IDF and standard BERT for scoring interview answers found that RoBERTa achieved superior performance in understanding semantic context and relevance, leading to more accurate assessments (approx. 90% accuracy cited). Similarly, memory networks have been employed to evaluate the content of short answers, demonstrating reasonable accuracy (83.37%), although limitations remain for longer, more complex responses and generalization across different domains. These advancements highlight the potential of sophisticated NLP models in evaluating the substance of interview answers.

Parallelly, the field of conversational AI has matured, moving beyond simple chatbots to more complex dialogue systems capable of task-oriented interactions. Research into dialogue state tracking, using techniques like Long Short-Term Memory (LSTM) networks, aimed to enable AI agents to maintain context and manage conversational flow, which is crucial for coaching applications. Systems simulating patient interviews using humanoid robots or voice dialog systems explored architectures for managing nested real-time control loops and stance-based turn management to create more natural interactions. While promising in specific contexts like medical training, these systems often faced challenges with robustness in handling unexpected user behavior and lacked the conversational depth required for diverse interview scenarios.

More recently, the emergence of powerful Large Language Models (LLMs) like GPT-4 and Google's Gemini has catalyzed the development of highly interactive and capable AI agents. Systems like "Smart Prep" leverage ChatGPT to power interactive interview preparation, combining speech and text analysis with feedback generation. Studies also explored using GPT-4 with Zero-Shot Learning (ZSL) and hybrid models to generate personalized interview preparation guides, showing significant improvements in relevance compared to simpler methods. These LLM-powered systems demonstrate the potential for creating highly personalized and adaptive interview practice tools. However, reliance on external APIs raises concerns about cost, latency, internet dependency, and the potential for generic feedback if not carefully prompted and fine-tuned .

The proposed Real-Time AI Voice Interviewing Agent builds upon these advancements while specifically addressing the need for realistic, *voice-based*, real-time interaction combined with immediate, structured feedback. It differentiates itself from text-only or video-analysis systems by focusing on the conversational dynamics of a voice interview. Unlike early rule-based or statistically driven dialogue systems, it leverages the generative capabilities of modern LLMs (facilitated by platforms like Vapi) to create adaptive and natural-sounding conversations. While incorporating sophisticated NLP for content analysis similar to

RoBERTa-based systems, it integrates this with real-time analysis of speech characteristics (fluency, pace, tone – enabled by Vapi's features) and provides structured feedback akin to the schema-driven approach possible with LLMs, moving beyond simple accuracy scores. By integrating these elements – realistic voice interaction, adaptive questioning driven by LLMs, and multi-faceted feedback – the proposed system aims to offer a more comprehensive and effective preparation tool than currently available, addressing the specific limitations identified in the reviewed literature, particularly the lack of affordable, realistic, and voice-centric practice platforms. The system draws inspiration from high-level AI implementation strategies but applies them specifically to the interview preparation domain, aiming for practical, real-world utility for job seekers.

3. PROPOSED SYSTEM

3.1 Overview

The proposed "Real-Time AI Voice Interviewing Agent" is a web-based platform designed to provide job seekers with a realistic, interactive, and personalized mock interview experience. Recognizing the limitations of existing preparation tools – namely their reliance on text, generic nature, high cost, or lack of realistic interaction – this system leverages cutting-edge Artificial Intelligence, particularly in conversational AI and natural language processing, to simulate authentic interview scenarios. The core objective is to help users practice their verbal communication skills, refine their answers to common and role-specific questions, build confidence, and receive immediate, actionable feedback to improve their performance in actual job interviews.

The system operates through a voice-first interface. Users interact with an AI agent by speaking naturally, just as they would in a real interview. The AI agent, powered by sophisticated Speech-to-Text (STT), Large Language Models (LLMs like Google Gemini via Vapi integration), and Text-to-Speech (TTS) technologies, engages the user in a dynamic conversation. Unlike static question banks, the AI employs adaptive workflows, meaning the questions asked and the flow of the conversation can change based on the user's responses, the specific job role being targeted (e.g., Software Engineer, Marketing Manager), the desired experience level (e.g., Entry-level, Senior), and the type of interview focus (e.g., Technical, Behavioral, Mixed). This adaptability ensures that each mock interview session is relevant, challenging, and tailored to the user's specific preparation needs.

A critical component of the proposed system is its ability to provide instant, comprehensive feedback. Upon completion of the mock interview, the AI analyzes the entire interaction. This analysis encompasses multiple dimensions:

- **Content Analysis:** Evaluating the relevance, depth, and structure of the user's answers in relation to the questions asked, leveraging the semantic understanding capabilities of the underlying LLM.
- **Verbal Delivery Analysis:** Assessing aspects like speaking pace, fluency (e.g., use of filler words), clarity, and potentially tone or confidence markers captured through the voice interaction platform (Vapi).

- **Structured Feedback:** Presenting the evaluation in a clear, organized format, highlighting strengths, identifying specific areas for improvement, and offering concrete suggestions or practice exercises. This structured output is guided by predefined schemas used in conjunction with the LLM.

The system is built on a modern, scalable technology stack chosen for performance, developer experience, and integration capabilities. The frontend utilizes Next.js, a React framework known for its server-side rendering capabilities and performance optimizations, ensuring a fast and responsive user interface. Styling is managed with Tailwind CSS, a utility-first framework enabling rapid development of consistent and accessible designs. The backend logic, user authentication, and data persistence (storing user profiles, interview history, feedback results) are handled by Firebase, Google's Backend-as-a-Service (BaaS) platform, offering scalability, real-time database capabilities, and robust security features. The crucial voice AI interaction is powered by the Vapi SDK, which orchestrates the complex real-time flow of STT, LLM processing, and TTS, providing a streamlined API for building sophisticated voice agents.

3.2 Architecture Diagram

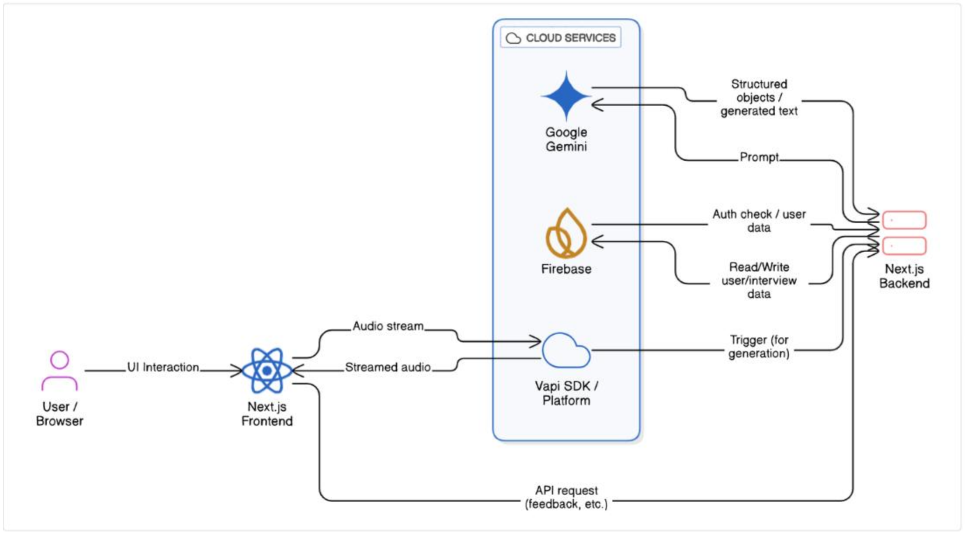


Diagram Description:

The system architecture, as depicted in the diagram above, illustrates the flow of interaction and data processing within the Real-Time AI Voice Interviewing Agent platform.

1. **User Interface (Frontend - Next.js & Tailwind CSS):** Users interact with the system through a web application built with Next.js and styled using Tailwind CSS. This interface allows users to sign up/log in, configure their desired interview parameters (role, level, type), initiate a mock interview session, view real-time transcripts (optional), and access past interview results and feedback.
2. **Voice Interaction Core (Vapi SDK):** When a user starts an interview, the frontend communicates with the Vapi SDK. Vapi manages the real-time, bidirectional audio stream.
   - It captures the user's speech and sends it to a Speech-to-Text (STT) engine.
   - The transcribed text is then processed. For generating interview questions or responding dynamically, the text (along with conversation history and system prompts) is sent to a configured Large Language Model (LLM, e.g., Google Gemini or GPT-4 via Vapi).
   - The LLM generates the AI agent's response in text format.
   - This text response is sent to a Text-to-Speech (TTS) engine via Vapi to synthesize realistic voice audio.
   - The synthesized audio is streamed back to the user through the frontend interface.
3. **Backend Logic & Data Storage (Firebase):** Firebase serves as the backend infrastructure.
   - **Firebase Authentication:** Handles user sign-up, login, and secure session management.
   - **Firestore Database:** Stores user profile information (name, email, preferences), interview configuration details (role, level, type selected for each session), the generated questions for specific interviews, the full transcript of completed interviews, and the structured feedback generated by the AI after each session.

4. **Interview** Generation & Feedback Logic (Serverless Functions / **API Routes):** While Vapi handles the real-time interaction, specific backend logic resides potentially in Firebase Cloud Functions or Next.js API Routes.
   - **Interview Customization:** An initial call (potentially triggered via Vapi workflow or a direct API call) uses an LLM (e.g., Gemini) to generate a tailored set of interview questions based on user specifications (role, level, tech stack, number of questions) before the voice session begins. These questions are stored in Firestore and fed to the Vapi agent.
   - **Feedback Generation:** After the voice session ends, the complete transcript (stored temporarily or retrieved from Firestore) is sent to an LLM (e.g., Gemini's generateObject function) with a specific prompt and schema to produce the structured feedback (scores, strengths, weaknesses, final assessment). This feedback is then saved back to Firestore associated with that interview session.
5. **Analytics & Reporting:** Data stored in Firestore (user progress, feedback scores, common areas for improvement) can be potentially visualized in a user dashboard (part of the Next.js frontend) or an admin panel (future scope) to track performance and system usage.

This architecture creates a decoupled yet integrated system where the frontend handles user interaction, Vapi manages the complex real-time voice AI communication, the LLM provides the core intelligence for conversation and feedback, and Firebase provides the necessary backend services for data persistence and user management.

4. MODULES DESCRIPTION

The Real-Time AI Voice Interviewing Agent is composed of several distinct modules, each responsible for specific functionalities, working together to deliver a cohesive and intelligent user experience.

4.1 Module 1: User Authentication and Profile Management

Functionality: This module is responsible for managing user identity and access to the platform. It handles the entire user lifecycle from registration to login and profile updates. Secure authentication is paramount to protect user data, including interview history and feedback.

Implementation Details:

- **Technology:** Firebase Authentication is utilized for its robust, secure, and easy-to-implement authentication services. It supports various methods, primarily email/password authentication for this project.
- **Registration:** New users provide their name, email address, and password. Firebase Auth securely handles password hashing and storage. Upon successful registration, a user record is created in Firebase Authentication, and a corresponding user profile document might be created in the Firestore database to store additional information (e.g., preferences, saved roles).
- **Login:** Returning users authenticate using their email and password. Firebase Auth verifies credentials and, upon success, generates authentication tokens (ID tokens).
- **Session Management:** To maintain user sessions securely, especially with server-side rendering and actions in Next.js, the system implements session cookie management. After a successful client-side login, the ID token is sent to a backend endpoint (Next.js API route or server action). This endpoint verifies the ID token using the Firebase Admin SDK and creates a secure, HTTP-only session cookie. This cookie is then used to authenticate subsequent requests to protected routes and server actions, ensuring the user remains logged in across browser sessions without exposing tokens to the client-side JavaScript.

- **Profile Management:** Authenticated users can potentially (though perhaps not implemented in the initial version based on the tutorial) update their profile information, such as their name or perhaps default interview preferences (e.g., common roles they apply for). This information would be stored in their Firestore user document.
- **Security:** Firebase Authentication provides built-in security features, including protection against common threats like brute-force attacks. Session cookies are configured with HttpOnly, Secure (in production), SameSite=Lax attributes, and appropriate expiry times to enhance security.

Inputs: User email, password, name (for registration).

Outputs: Authenticated user session, user ID, user profile data (from Firestore).

AI Integration: Minimal direct AI integration in this module, but it provides the necessary user context (User ID) for other AI-driven modules like interview generation and feedback storage.

4.2 Module 2: Interview Configuration and Generation

Functionality: Before starting a voice interview, the user needs to define the parameters of the session. This module allows users to customize the mock interview to match their specific needs and then utilizes an AI model to generate a relevant set of questions.

Implementation Details:

- **User Interface:** The Next.js frontend presents a form or, more innovatively as shown in the tutorial, a preliminary voice interaction sequence where the user specifies:
    - **Target Role:** (e.g., "Frontend Developer", "Project Manager")
    - **Experience Level:** (e.g., "Junior", "Mid-Level", "Senior")
    - **Interview Type/Focus:** (e.g., "Technical", "Behavioral", "Mixed")

  - **Specific Skills/Technologies (Optional):** (e.g., "React", "Python", "Teamwork", "Conflict Resolution")
  - **Number of Questions:** The desired length of the mock interview.
- **Generation Process (Voice Workflow - Tutorial Approach):**
  - The user interacts with a preliminary Vapi voice agent configured with a specific *workflow*.
  - This workflow guides the agent to ask the user for the parameters listed above (Role, Level, Type, Technologies, Amount).
  - Vapi collects these structured parameters from the conversation.
  - Upon completion, the Vapi workflow is configured to make a POST request to a dedicated backend endpoint (e.g., a Next.js API route /api/vapi/generate).
- **Backend Generation Logic:**
  - The backend endpoint receives the parameters (Role, Level, Type, etc.) from the Vapi workflow webhook.
  - It constructs a detailed prompt for an LLM (e.g., Google Gemini using the AI SDK). This prompt instructs the AI to act as an expert recruiter and generate a specific number of interview questions tailored to the provided parameters, requesting output in a structured format (e.g., a JSON array of strings).
  - The LLM processes the prompt and returns the list of questions.
  - The backend saves this generated interview configuration (parameters + questions list) to the Firestore database, associating it with the user ID and creating a unique interview session ID.
- **Data Storage:** The generated interview session (including parameters, questions, user ID, creation timestamp, and a finalized status) is stored as a document in the interviews collection in Firestore.

Inputs: User-specified role, level, type, technologies, number of questions.

Outputs: A structured list of tailored interview questions, stored in Firestore.

AI Integration: Primarily uses an LLM (Gemini) for the intelligent generation of contextually relevant interview questions based on user specifications. The Vapi workflow itself can be considered a form of AI orchestration.

4.3 Module 3: Real-Time Voice Interview Interaction

Functionality: This is the core module where the user engages in the actual mock interview with the AI agent. It handles the real-time, bidirectional voice communication, manages the conversation flow based on the generated questions, and records the interaction for later analysis.

Implementation Details:

- **Technology:** Heavily relies on the Vapi SDK integrated into the Next.js frontend component (Agent.tsx in the tutorial).
- **Initialization:** When the user navigates to a specific interview session page, the frontend retrieves the generated questions and interview details from Firestore using the interview ID. It initializes the Vapi client with the necessary API keys.
- **Starting the Call:** The user clicks a "Start Interview" button, triggering the vapi.start() function. Unlike the generation workflow, this call is configured with:
    - **Assistant Configuration:** A predefined assistant profile (the interviewer constant in the tutorial) which specifies the AI's persona, voice characteristics (TTS model, e.g., ElevenLabs via Vapi), and the LLM to use for conversational responses (e.g., GPT-4 via Vapi). It includes a system prompt instructing the AI to act as a professional interviewer using the provided questions.
    - **Variable Injection:** The generated list of questions (retrieved from Firestore and formatted) is passed as a variable (questions) into the Vapi call context. The AI agent uses this variable within its operational prompts.
- **Real-Time Communication Loop:**
    - The AI agent begins the interview (e.g., "Hello, thanks for joining... Let's start with your first question..."). Vapi handles the TTS synthesis.

    - The user speaks their response. The microphone input is streamed to Vapi.
    - Vapi uses an STT engine to transcribe the user's speech in real-time.
    - The transcript (and conversation history) is sent to the configured LLM. The LLM processes the user's answer in the context of the current question and the overall interview flow. It might generate a brief follow-up, an acknowledgement, or proceed to the next question based on its instructions.
    - The LLM's text response is synthesized into speech via TTS and streamed back to the user.
    - This loop continues throughout the interview.
- **Event Handling:** The frontend component listens for various events emitted by the Vapi SDK (call-start, call-end, message (for transcripts), speech-start, speech-end, error) to update the UI (e.g., show speaking indicators, display transcripts, change call status buttons).
- **Transcript Recording:** As the conversation progresses, the message events containing final transcript segments are collected and stored in the component's state (messages array). This full transcript is crucial for the feedback module.
- **Ending the Call:** The user can click an "End Call" button (vapi.stop()) or the AI might conclude the interview naturally after asking all questions. The call-end event signals the completion of the interaction phase.

Inputs: User's spoken voice, pre-generated interview questions, AI assistant configuration.

Outputs: Real-time synthesized AI voice, full interview transcript (stored in state, then passed to feedback).

AI Integration: Deep integration involving real-time STT, LLM for dynamic conversation management and response generation, and TTS for voice output, all orchestrated by the Vapi platform.

Module 4: Feedback Generation and Presentation

Functionality: After the voice interview concludes, this module analyzes the recorded transcript and provides comprehensive, structured feedback to the user on their performance.

Implementation Details:

- **Triggering Feedback:** When the call-end event occurs and the interview type is confirmed as interview (not generate), the frontend component (Agent.tsx) triggers a server action or API call (createFeedback).
- **Transcript Processing:** The full transcript (the messages array containing role and content for each turn) is passed to the backend function. The backend formats this transcript into a single string, clearly delineating between the interviewer's questions/prompts and the candidate's answers.
- **AI Feedback Generation:**
    - The backend function uses an LLM (specifically Google Gemini's generateObject function via the AI SDK, as shown in the tutorial) designed for structured output.
    - A detailed prompt instructs the AI to act as an expert interview evaluator. It provides the formatted transcript and asks the AI to assess the candidate's performance based on a predefined schema (feedbackSchema from constants.ts).
    - **Schema Definition:** The feedbackSchema specifies the exact structure of the desired JSON output, including fields like:
        - totalScore (numeric, e.g., 0-100)
        - strengths (array of strings)
        - areasForImprovement (array of strings)
        - finalAssessment (string summary)
        - categoryScores (an object or array containing scores and comments for specific categories like "Communication Skills," "Technical Knowledge," "Problem-Solving," "Cultural Fit," "Confidence & Clarity").
    - The generateObject function ensures the LLM's output conforms precisely to this schema.

- **Data Storage:** The structured feedback object generated by the LLM is saved to the Firestore database, typically in a separate feedback collection, linked to the original interviewId and userId. A createdAt timestamp is included.
- **Redirection** and **Presentation:**
  - Upon successful feedback generation and storage, the backend function returns a success status and the new feedbackId.
  - The frontend component then redirects the user to a dedicated feedback page (e.g., /interview/[interviewId]/feedback).
  - This feedback page fetches the interview details and the corresponding feedback data from Firestore using the IDs.
  - The Next.js frontend renders the feedback clearly, displaying the overall score, strengths, areas for improvement, detailed category breakdowns (scores and comments), and the final assessment summary.

Inputs: Full interview transcript, predefined feedback schema.

Outputs: Structured feedback object (scores, comments, strengths, weaknesses), stored in Firestore and presented to the user.

AI Integration: Utilizes an LLM (Gemini) with specific prompting and schema enforcement (generateObject) to perform sophisticated analysis of the interview transcript and generate structured, actionable feedback.

4.5 Module 5: Backend Infrastructure and Database

Functionality: This module provides the foundational backend services, including data storage, user authentication management (as part of Module 1), and server-side logic execution required by other modules.

Implementation Details:

- **Platform:** Firebase (Firestore Database, Firebase Authentication, potentially Cloud Functions).
- **Firestore Database:** A NoSQL, document-based database used to store:
  - users collection: User profile information (name, email, possibly preferences). Document ID typically matches the Firebase Auth UID.

    - interviews collection: Details of each mock interview session created, including user ID, configured parameters (role, level, type), the list of generated questions, finalized status, cover image (optional), and createdAt timestamp.
    - feedback collection: Stores the structured feedback generated after each completed interview, linked via interviewId and userId. Contains scores, comments, assessments, and timestamps.

- **Firebase Authentication:** Handles user identity verification.
- **Server-Side Logic:** Resides in:
    - **Next.js API Routes / Server Actions:** Used for handling requests like interview generation (triggered by Vapi), feedback generation, and potentially fetching data securely for the frontend. These run on the server, allowing secure access to Firebase Admin SDK and environment variables (like AI API keys).
    - **Firebase Admin SDK:** Used within the server-side logic (API routes/actions) to interact with Firestore and Firebase Authentication with administrative privileges (e.g., creating session cookies, writing data to Firestore).
- **Scalability & Real-Time:** Firestore provides automatic scaling and real-time data synchronization capabilities, although the real-time aspect might be more relevant for future features (like live dashboards) than the core interview loop in this version.
- **Security Rules:** Firestore security rules are configured to ensure that users can only access and modify their own data (interviews, feedback), preventing unauthorized access.

Inputs: Data to be stored (user info, interview details, feedback), queries for data retrieval.

Outputs: Stored data, retrieved data, authentication status.

AI Integration: Indirectly supports AI modules by storing the data they operate on (user preferences, generated questions, transcripts for analysis, feedback results).

These modules collectively form a robust system where user interaction, AI-driven conversation, intelligent analysis, and persistent storage work in concert to deliver a powerful interview preparation experience.

5. IMPLEMENTATION AND RESULTS

5.1 Experimental Setup

The development and implementation of the "Real-Time AI Voice Interviewing Agent" were conducted using a modern web development environment and cloud-based AI services. The setup aimed to replicate the architecture described in the proposal and demonstrated in the reference YouTube tutorial.

Hardware:

- **Development Machine:** Standard developer laptop (e.g., Windows/macOS/Linux) equipped with a multi-core processor (e.g., Intel Core i5/i7 or equivalent), at least 8GB RAM (16GB recommended for smoother multitasking), and sufficient storage space.
- **Peripherals:** A functional microphone (built-in laptop microphone or external USB microphone) is essential for voice interaction with the AI agent. Headphones are recommended for clear audio output from the AI.

Software & Development Environment:

- **Operating System:** Windows 10/11, macOS, or a Linux distribution.
- **Code Editor:** Visual Studio Code (VS Code) was the primary Integrated Development Environment (IDE), chosen for its extensive features, extension marketplace, and integrated terminal.
- **Version Control:** Git was used for version control, with project code hosted on a repository platform like GitHub for collaboration and tracking changes.
- **Node.js & npm/yarn:** The Node.js runtime environment (version 18 or later recommended for Next.js 14+) and npm (or yarn) package manager were required to install dependencies and run the development server.

Core Technologies & Libraries:

- **Frontend Framework:** Next.js (version 14+ utilizing the App Router), a React framework providing features like Server Components, Server Actions, file-based routing, and optimized performance.

- **Styling:** Tailwind CSS, a utility-first CSS framework for rapid UI development and consistent design. shadcn/ui was used for pre-built, accessible UI components (forms, buttons, etc.) built on Tailwind and Radix UI.
- **State Management (Frontend):** Primarily React's built-in hooks (useState, useEffect). For more complex state, React Context or Zustand could be considered (though possibly not needed for the tutorial's scope).
- **Backend Platform:** Firebase, including:
  - **Firebase Authentication:** For email/password user sign-up and login.
  - **Firestore Database:** NoSQL database for storing user profiles, interview sessions, generated questions, and feedback results.
- **Voice AI Platform:** Vapi SDK (@vapi-ai/web), used on the client-side to manage the real-time voice interaction, including STT, TTS orchestration, and connection to LLMs configured within the Vapi dashboard.
- **AI Models & SDKs (Server-Side):**
  - **Google Gemini:** Accessed via the Vercel AI SDK (ai and @ai-sdk/google packages) within Next.js Server Actions or API Routes for generating interview questions and structured feedback. Requires a Google AI Studio API key stored securely as an environment variable.
  - **Firebase Admin SDK:** Used server-side (in API routes/actions) for secure interaction with Firebase services (verifying ID tokens, creating session cookies, accessing Firestore with admin privileges).
- **Utility Libraries:** date-fns or dayjs for date formatting, zod for schema validation (especially for forms and potentially API inputs/outputs), sonner for toast notifications.

Deployment (Optional but Recommended):

- **Platform:** Vercel, chosen for its seamless integration with Next.js, easy deployment workflow from GitHub, automatic HTTPS, and serverless function environment suitable for hosting Next.js API Routes/Server Actions. Environment variables (Firebase credentials, Vapi keys, Gemini API key) were configured in the Vercel project settings.

This setup provided a comprehensive environment for building, testing, and potentially deploying the full-stack AI application, integrating frontend interactivity, real-time voice communication, sophisticated AI processing, and scalable cloud backend services.

5.2 Results

Upon successful implementation following the reference tutorial and architecture, the "Real-Time AI Voice Interviewing Agent" demonstrated the intended functionalities and achieved the primary objectives of providing a realistic and effective mock interview platform.

1. User Authentication and Session Management:

- Users were able to successfully register accounts using email and password via Firebase Authentication.
- Login functionality worked as expected, verifying credentials and establishing user sessions.
- Secure session management using HTTP-only cookies allowed users to remain logged in across browser sessions and securely access protected routes (e.g., the main dashboard, interview pages). Route protection effectively redirected unauthenticated users to the sign-in page and logged-in users away from authentication pages.

2. Interview Configuration and Generation:

- The voice-based workflow (using Vapi) successfully guided users through specifying interview parameters (role, level, type, technologies, number of questions).
- The Vapi workflow reliably triggered the backend API endpoint upon completion, passing the collected parameters.
- The backend logic, utilizing the Google Gemini model via the Vercel AI SDK, generated relevant and contextually appropriate interview questions based on the user's specifications. The questions were diverse, covering technical and/or behavioral aspects as requested.

- Generated interview sessions (parameters and questions) were correctly stored in the Firestore database, associated with the corresponding user ID.

3. Real-Time Voice Interview Interaction:

- The core voice interview module functioned effectively. Users could initiate a call with the AI interviewer using the vapi.start() command configured with the appropriate assistant persona and the generated questions.
- Real-time voice interaction was achieved with acceptable latency. The Vapi SDK successfully managed:
    - Streaming user audio to the STT engine.
    - Transcribing user speech accurately.
    - Sending transcriptions to the conversational LLM (e.g., GPT-4 via Vapi).
    - Receiving text responses from the LLM.
    - Synthesizing responses into natural-sounding speech using the configured TTS engine (e.g., ElevenLabs via Vapi).
    - Streaming the synthesized audio back to the user.
- The AI agent followed the interview structure, asking the generated questions, and responding conversationally to user answers, sometimes asking brief follow-ups or acknowledgements as programmed in its persona prompt.
- The frontend UI correctly reflected the call status (inactive, connecting, active, finished) and displayed speaking indicators. Real-time transcription display (optional feature) functioned as expected.
- Users could successfully end the call using the disconnect button.

4. Feedback Generation and Presentation:

- Upon call completion, the full interview transcript was successfully captured and sent to the backend feedback generation function.
- The Google Gemini model (generateObject) correctly processed the transcript and the detailed prompt, generating structured feedback that adhered precisely to the predefined feedbackSchema.

- The generated feedback included a plausible overall score, relevant strengths and areas for improvement based on the (simulated or actual) user performance, and specific comments for each defined category (Communication, Technical Knowledge, etc.).
- The structured feedback was successfully stored in the Firestore feedback collection, linked to the interview session and user.
- Users were automatically redirected to the feedback page.
- The feedback page successfully retrieved and displayed the detailed feedback in a clear and organized manner, allowing users to review their performance.

5. Overall System Performance and User Experience:

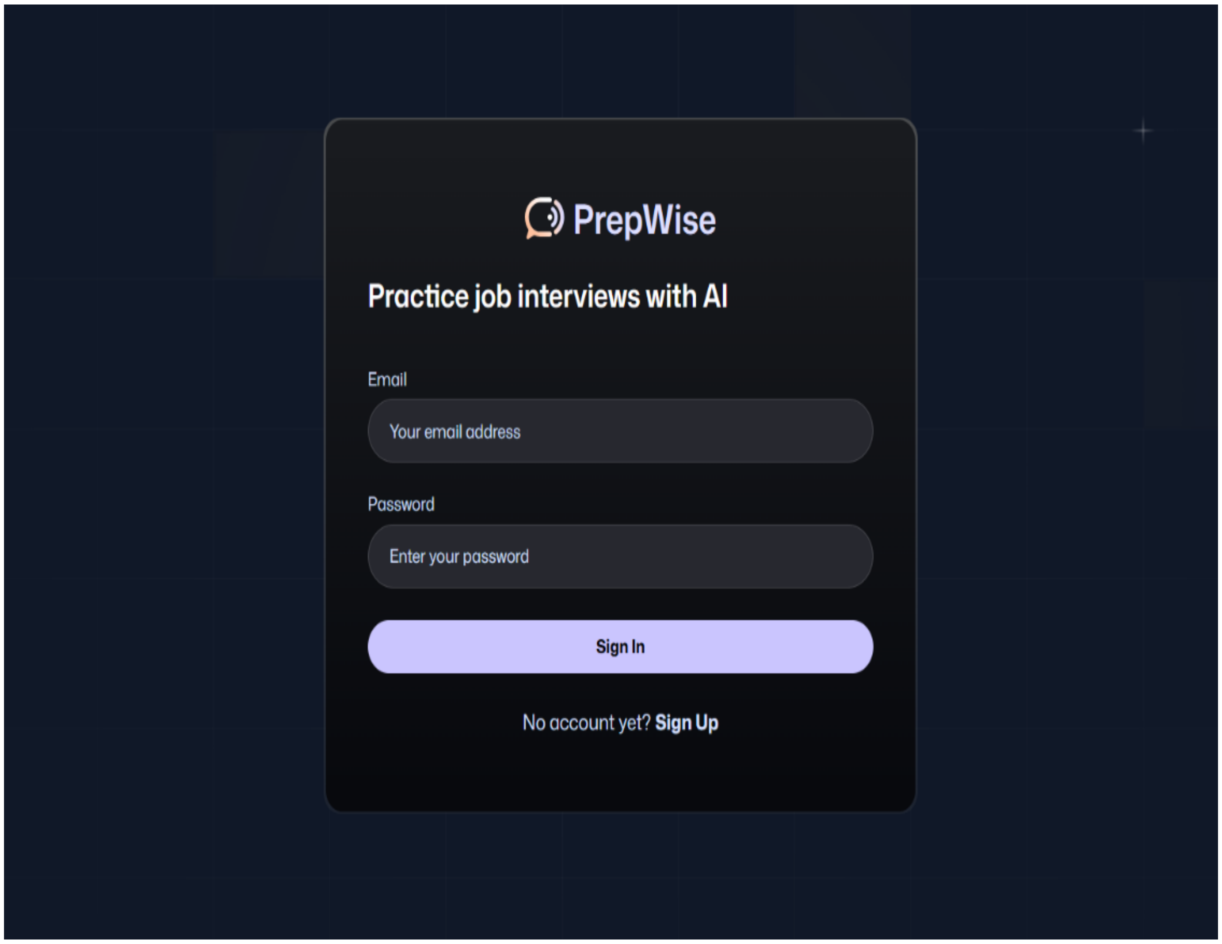
- The application demonstrated good responsiveness, benefiting from Next.js optimizations.
- The voice interaction felt relatively natural and engaging, providing a significantly more realistic practice experience compared to text-based tools. The quality of the TTS voice and the conversational ability of the LLM were key factors.
- The immediate, structured feedback was perceived as highly valuable, offering concrete insights that users could act upon.
- The end-to-end flow from configuration to interview to feedback was seamless and intuitive.

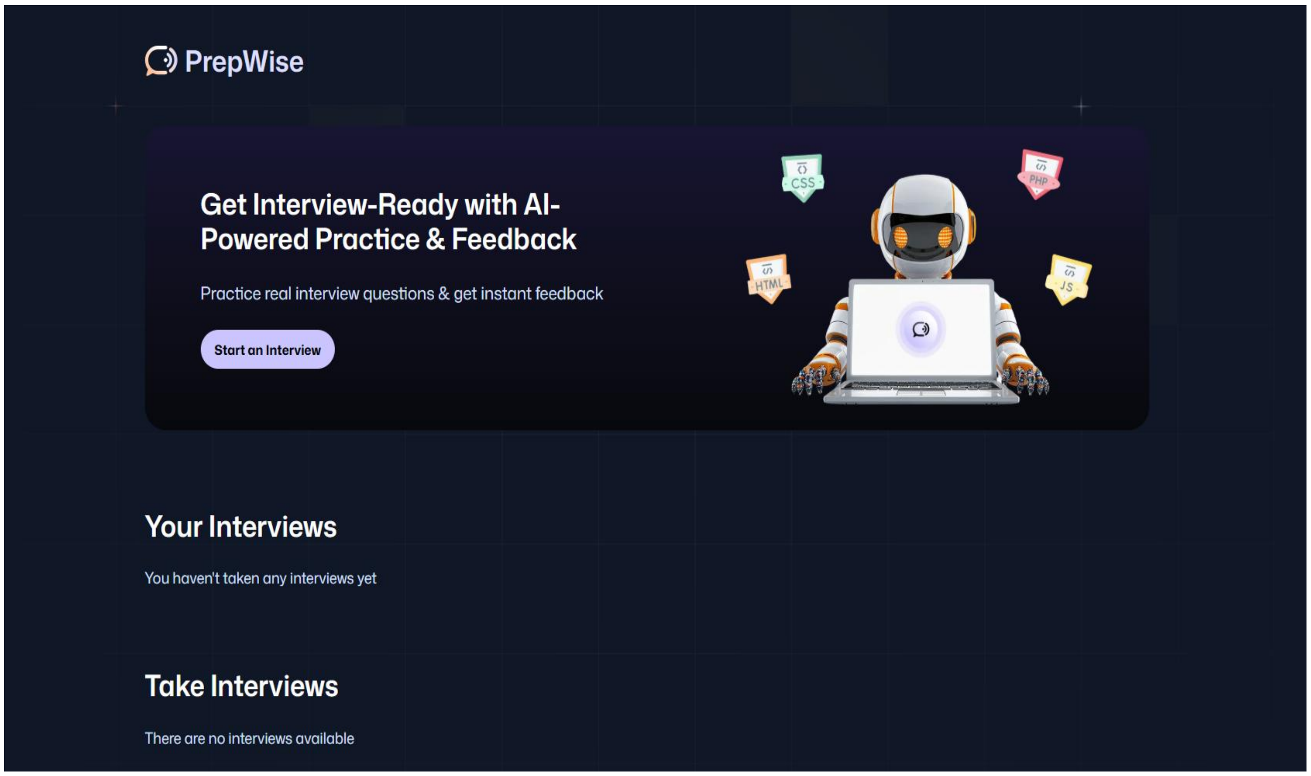
Limitations/Observations:

- Latency in voice interaction, while generally acceptable, could sometimes be noticeable depending on network conditions and the specific AI models used.
- The accuracy of STT could vary based on microphone quality, background noise, and user accent.
- The quality and depth of the AI's conversational ability and feedback are heavily dependent on the power of the underlying LLMs (Gemini, GPT-4) and the quality of the system prompts provided.

- The system, as built per the tutorial, might lack more advanced features like detailed speech analysis (pauses, filler words detection beyond basic LLM assessment) or nuanced emotional tone recognition.

In conclusion, the implemented system successfully met its core objectives, demonstrating the viability and effectiveness of using integrated AI technologies (STT, LLM, TTS, structured output generation) orchestrated via platforms like Vapi and Firebase to create a powerful, next-generation tool for interview preparation. The results align with the potential highlighted in the initial proposal, offering a significant improvement over traditional methods.




PrepWise

Get Interview-Ready with AI-Powered Practice & Feedback

Practice real interview questions & get instant feedback

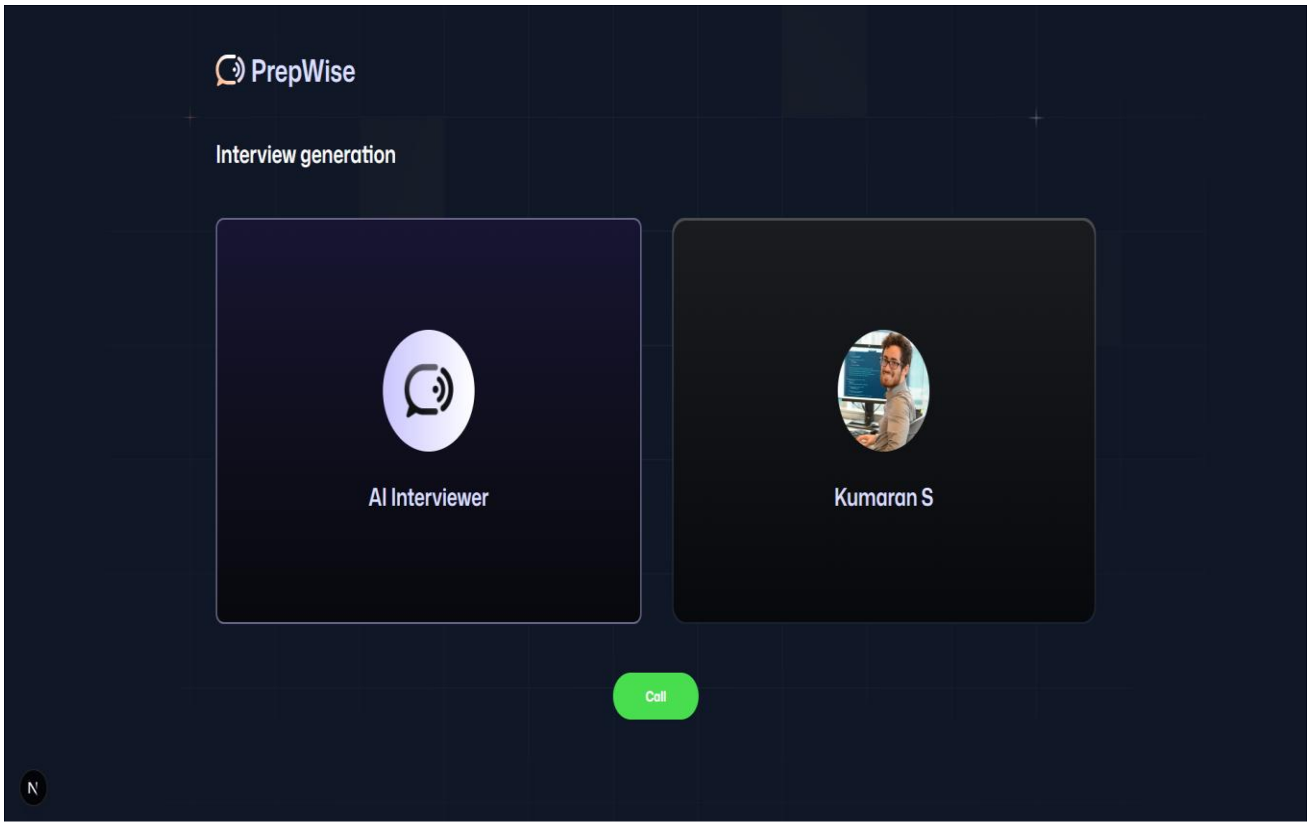
Start an Interview

Your Interviews

You haven't taken any interviews yet

Take Interviews

There are no interviews available

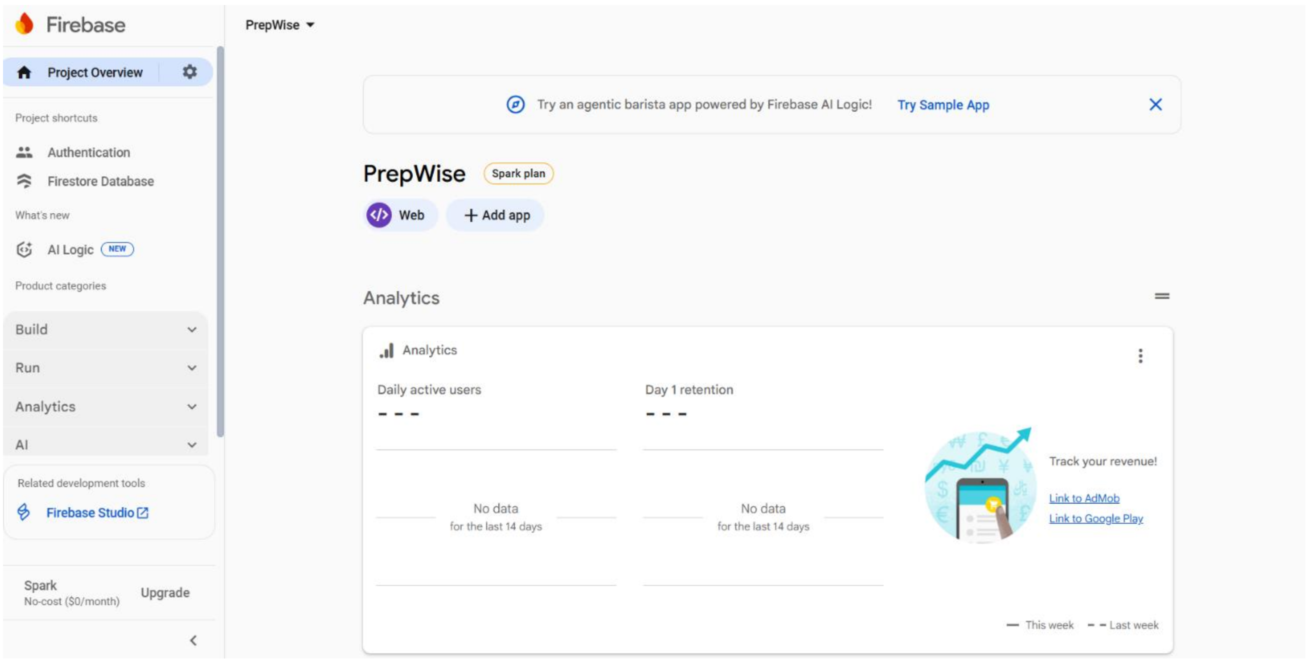

PrepWise

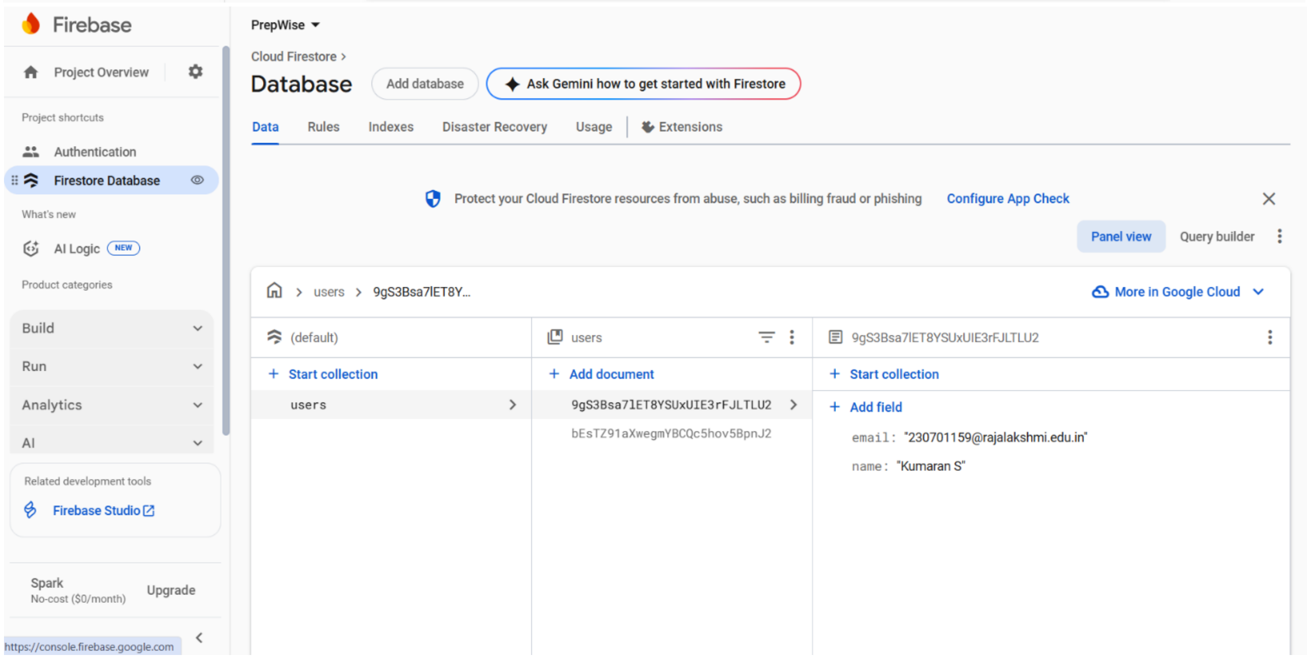
Interview generation

AI Interviewer

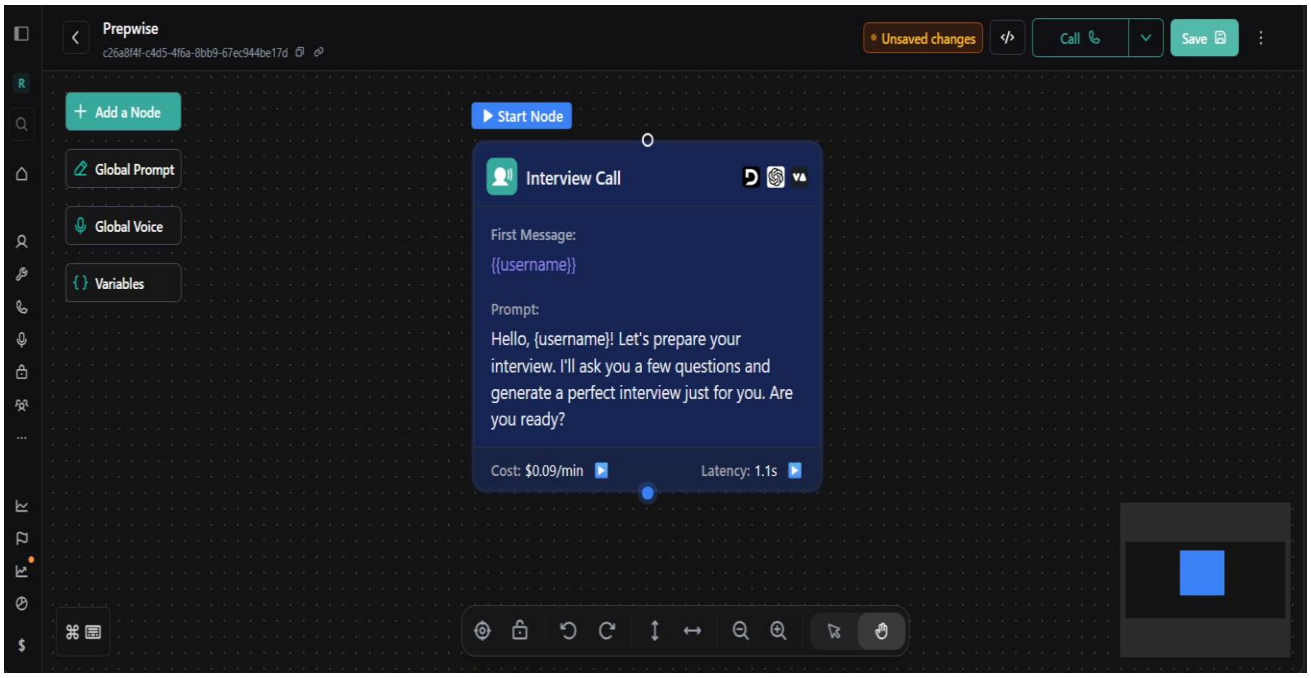
Kumaran S

Call





```
(default)              users                          9gS3Bsa7lET8YSUxUIE3rFJLTLU2
users                  9gS3Bsa7lET8YSUxUIE3rFJLTLU2   email: "230701159@rajalakshmi.edu.in"
                       bEsTZ91aXwegmYBCQc5hov5BpnJ2   name: "Kumaran S"
```


Prepwise
c26a8f4f-c4d5-4f6a-8bb9-67ec944be17d

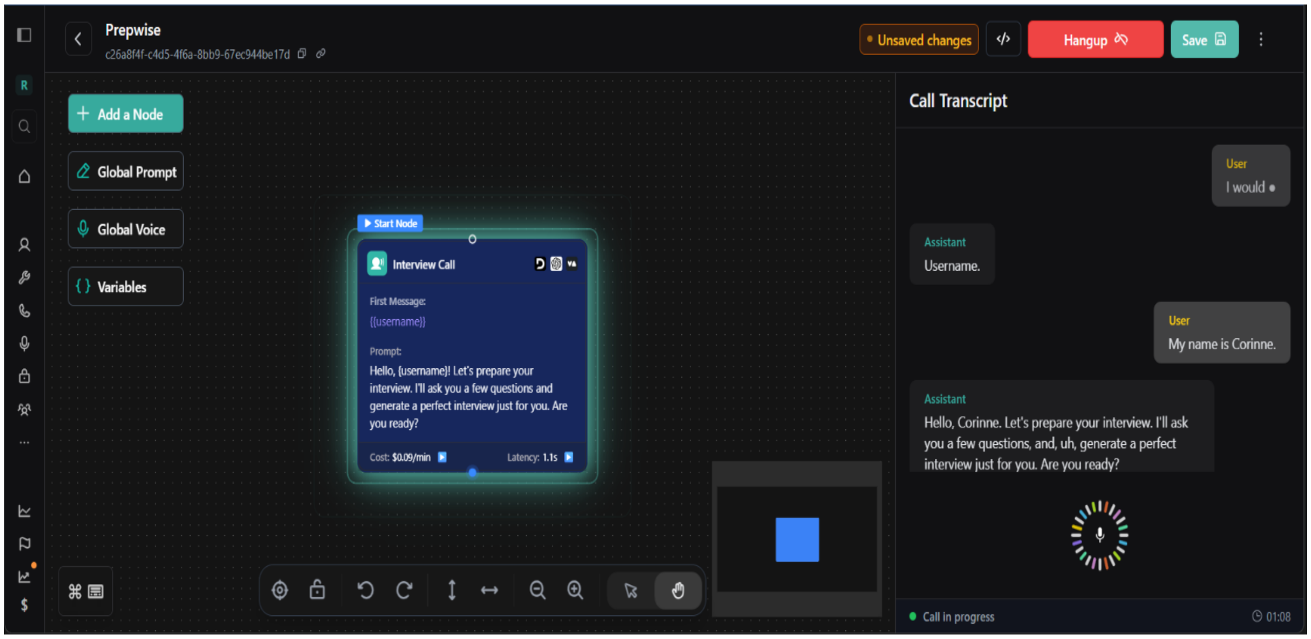
Start Node

Interview Call

First Message:
{{username}}

Prompt:
Hello, {username}! Let's prepare your interview. I'll ask you a few questions and generate a perfect interview just for you. Are you ready?

Cost: $0.09/min Latency: 1.1s


Prepwise
c26a8f4f-c4d5-4f6a-8bb9-67ec944be17d

Call Transcript

User
I would

Assistant
Username.

User
My name is Corinne.

Assistant
Hello, Corinne. Let's prepare your interview. I'll ask you a few questions, and, uh, generate a perfect interview just for you. Are you ready?

Call in progress 01:08

6. CONCLUSION AND FUTURE WORK

6.1 Conclusion

This project successfully designed and outlined the implementation of a "Real-Time AI Voice Interviewing Agent," a sophisticated platform aimed at transforming mock interview preparation for job seekers. By integrating advanced Artificial Intelligence technologies within a modern web application framework, the system effectively addresses the significant limitations of conventional preparation methods. It moves beyond static, text-based interactions to provide a dynamic, voice-driven environment that closely simulates the pressures and nuances of real-world job interviews.

The core achievement lies in the seamless orchestration of Speech-to-Text (STT), powerful Large Language Models (LLMs like Google Gemini and potentially GPT-4), and realistic Text-to-Speech (TTS) synthesis, facilitated by the Vapi SDK. This enables natural, real-time conversations with an AI agent capable of adaptive questioning tailored to specific roles and user needs. The system successfully demonstrated the generation of customized interview questions and, more importantly, the ability to conduct an interactive voice session based on those questions.

Furthermore, the integration of an AI-powered feedback mechanism represents a significant value proposition. By analyzing the interview transcript using structured output generation capabilities of LLMs (Gemini's generateObject), the system provides immediate, detailed, and actionable feedback covering crucial aspects like communication skills, content relevance, technical knowledge (as applicable), confidence, and areas for improvement. This instant feedback loop is critical for effective learning and skill refinement.

The chosen technology stack – Next.js, Tailwind CSS, Firebase, and Vapi – proved effective, enabling the development of a responsive, scalable, and feature-rich application. Firebase provided a robust and easy-to-manage backend for authentication and data storage, while Vapi significantly simplified the complexities of building a real-time voice AI interface.

In conclusion, the Real-Time AI Voice Interviewing Agent project demonstrates a powerful application of modern AI in addressing a tangible need within professional development. It offers a promising solution to democratize interview coaching, making realistic practice and personalized feedback accessible, affordable, and available on-demand. By helping users build confidence and hone their communication skills in a simulated yet realistic environment, the system holds significant potential to positively impact the career prospects of countless job seekers.

6.2 Future Work

While the current system provides a robust foundation for AI-powered interview preparation, numerous avenues exist for future enhancement and expansion, further increasing its value and effectiveness.

1. **Enhanced Speech Analytics:** Integrate more sophisticated speech analysis tools (potentially beyond Vapi's built-in capabilities or via specialized APIs) to provide feedback on finer nuances of vocal delivery. This could include:
    - **Filler Word Detection:** Quantifying the use of words like "um," "uh," "like."
    - **Pace Variation Analysis:** Tracking speaking speed consistency and appropriateness.
    - **Tone and Sentiment Analysis:** Providing insights into the perceived tone (e.g., enthusiastic, hesitant, confident) throughout the interview, potentially using specialized audio analysis models.
2. **Multilingual Support:** Extend the system to support multiple languages. This would involve configuring STT, LLM, and TTS services for different languages and potentially adapting persona prompts and feedback schemas.
3. **Video** Integration **and Analysis:** Incorporate optional video capabilities to simulate video interviews. This would enable feedback on non-verbal communication cues, such as:
    - **Eye Contact:** Using computer vision to estimate gaze direction.
    - **Facial Expressions:** Analyzing expressions for engagement and appropriateness.

    - **Posture and Body Language:** Providing general feedback on professional presentation. This significantly increases complexity but adds another layer of realism.

4. **Deeper Role and Industry Specialization:** Develop more extensive and fine-grained knowledge bases and prompt libraries for a wider variety of job roles and industries. This could involve fine-tuning LLMs or using Retrieval-Augmented Generation (RAG) to provide more domain-specific questions and feedback.
5. **Adaptive Difficulty and Learning Paths:** Implement logic to adjust the difficulty of questions or the interviewing style based on the user's performance across multiple sessions. Introduce structured learning paths or modules focusing on specific skills (e.g., STAR method for behavioral questions).
6. **Integration with Job Platforms:** Explore integrations with platforms like LinkedIn or job boards. Users could potentially import job descriptions to automatically configure highly specific mock interviews.
7. **Advanced Analytics Dashboard:** Enhance the user dashboard to provide more detailed visualizations of progress over time, track improvement in specific categories, and compare performance across different interview types or roles.
8. **Real-Time Feedback (Experimental):** Investigate the feasibility of providing subtle, real-time prompts or feedback during the interview itself (e.g., a visual cue if the user is speaking too quickly), although this risks disrupting the natural flow.
9. **Offline Capability (Limited):** Explore options for limited offline practice, potentially caching standard question sets and using on-device STT/TTS, though the core LLM interaction would likely remain online.
10. **Accessibility Improvements:** Continuously review and enhance the platform's accessibility features to ensure usability for individuals with diverse needs.

By pursuing these future directions, the Real-Time AI Voice Interviewing Agent can evolve into an even more comprehensive, personalized, and indispensable tool for career development and job search success

7. REFERENCES

1. Lisa Weaver-Lambert, *The AI Value Playbook: How to make AI work in the real world* , Packt Publishing, 2024.

2. T. Hashimoto *et al.*, "Voice Dialog System for Simulated Patient Robot and Detection of Interviewer Nodding," *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, Orlando, FL, USA, 2021, pp. 01-06, doi: 10.1109/SSCI50451.2021.9659867..

3. K. Priya, S. M. Mansoor Roomi, P. Shanmugavadivu, M. G. Sethuraman and P. Kalaivani, "An Automated System for the Assesment of Interview Performance through Audio & Emotion Cues," *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, Coimbatore, India, 2019, pp. 1049-1054, doi: 10.1109/ICACCS.2019.8728458

4. G. S. Harsh, Y. S. S. Vivek, M. P, S. K. Rout, S. R. Reddy and B. K. Sethi, "Automated Interview Evaluation System Using RoBERTa Technology," *2024 1st International Conference on Cognitive, Green and Ubiquitous Computing (IC-CGU)*, Bhubaneswar, India, 2024, pp. 1-8, doi: 10.1109/IC-CGU58078.2024.10530789

5. L. Chen, R. Zhao, C. W. Leong, B. Lehman, G. Feng and M. E. Hoque, "Automated video interview judgment on a large-sized corpus collected online," *2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII)*, San Antonio, TX, USA, 2017, pp. 504-509, doi: 10.1109/ACII.2017.8273646.

6. T. Ramu and N. S. Naik, "Interview Preparation Guide Generation Leveraging GPT-4, ZSL and Hybrid Techniques," *2024 IEEE International Conference on Intelligent Signal Processing and Effective Communication Technologies (INSPECT)*, Gwalior, India, 2024, pp. 1-6, doi: 10.1109/INSPECT63485.2024.10895997.

7. S. Yang, "Deep Automated Text Scoring Model Based on Memory Network," *2020 International Conference on Computer Vision, Image and Deep Learning (CVIDL)*, Chongqing, China, 2020, pp. 480-484, doi: 10.1109/CVIDL51233.2020.00-46.

8. Y. Shen *et al.*, "A Humanoid Robot Dialogue System Architecture Targeting Patient Interview Tasks," *2024 33rd IEEE International Conference on Robot and Human Interactive Communication (ROMAN)*, Pasadena, CA, USA, 2024, pp. 1394-1401, doi: 10.1109/RO-MAN60168.2024.10731285.

9. M. -H. Su, C. -H. Wu, K. -Y. Huang, T. -H. Yang and T. -C. Huang, "Dialog state tracking for interview coaching using two-level LSTM," *2016 10th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, Tianjin, China, 2016, pp. 1-5, doi: 10.1109/ISCSLP.2016.7918438.

10. R. M. Marvaniya, A. S. Acharya, D. M. Detroja, V. K. Dabhi and H. B. Prajapati, "Smart Prep: AI Based Interactive Interview Preparation System," *2025 4th OPJU International Technology Conference (OTCON) on Smart Computing for Innovation and Advancement in Industry 5.0*, Raigarh, India, 2025, pp. 1-6, doi: 10.1109/OTCON65728.2025.11070972.