AI Mock Interviewer

Archana Burujwale[1], Nikita Waghmare[2,] Harshada Mahesh Dhobale[3], Siddhesh Sabnis[4], Kaushtubh Lanke[5]
Department of Computer Science Engineering, Vishwakarma Institute of Technology, Pune, India

***Abstract*—In today's highly competitive job market, excelling in technical interviews has become crucial for candidates aspiring to secure positions in top companies. However, access to personalized interview preparation tools remains limited, especially those that provide real-time, meaningful feedback. This paper presents the design and development of an AI-powered mock interview platform that aims to bridge this gap by leveraging cutting-edge web technologies and generative AI. The platform is built using a full-stack architecture comprising React and Next.js for a dynamic frontend experience, Clerk for robust user authentication, Drizzle ORM with PostgreSQL for efficient backend data handling, and Gemini AI for generating domain-specific interview questions in real-time. Users can upload their resumes (optional), select their desired job role or interview category, and begin a live interview session where AI-generated questions are posed. Candidate responses are recorded via webcam and analysed using AI algorithms to assess various performance metrics, including tone, fluency, relevance, and confidence. A unique feature of this system is its automated feedback generation module, which provides actionable insights and personalized analytics to help users improve. The feedback is visualized through an interactive performance dashboard, making it easier for users to track progress over time. With scalability and security integrated into its core design, this platform caters to a wide user base including students, fresh graduates, and working professionals preparing for interviews. The proposed system not only enhances accessibility to highquality interview preparation tools but also provides a cost-effective and intelligent alternative to traditional coaching methods. This research highlights the technical architecture, implementation challenges, and evaluation of the platform's effectiveness, offering a blueprint for future developments in AI-driven educational tools. give in few ines in same format**

***Index Terms*—Mock Interview, Next.js, React, AI Interviewer, Gemini AI, Full-Stack Development, Drizzle ORM, PostgreSQL, Video Feedback System**

I. INTRODUCTION

The rise of generative AI, modern web frameworks, and cloud-native development offers a unique opportunity to reimagine interview preparation through scalable, intelligent, and interactive digital solutions. By combining the power of In today's fast-paced, competitive job market—particularly in artificial intelligence with full-stack web development, it is the field of technology—candidates are expected to demonstrate possible to create platforms that not only mimic real interview
not only technical expertise but also strong communication, conditions but also dynamically adapt to user input and provide problem-solving, and interpersonal skills during interviews. The personalized, actionable feedback. interview process has evolved to become more rigorous, with employers employing various techniques to evaluate both the
This research introduces a comprehensive, full-stack AI-
hard and soft skills of applicants. This has intensified the need powered mock interview platform that aims to address the for effective, consistent, and personalized interview preparation shortcomings of traditional methods. The system allows users to tools that can simulate real-world conditions and offer register and authenticate using Clerk, a secure and developer-meaningful feedback. friendly authentication service. Upon login, users can optionally upload their resume, select a job role or interview domain, and Traditional mock interviews conducted by career coaches or start a simulated interview. The platform uses Gemini AI to peers are useful but come with several limitations. These include generate context-aware interview questions based on the unavailability of skilled interviewers, inconsistent evaluation selected job profile or user resume data. Interviews are criteria, human biases,

time constraints, and high costs. conducted through a video interface, and user responses are Moreover, candidates may not always have access to expert recorded and temporarily stored for analysis. Post-interview, the feedback or may find it difficult to replicate realistic interview AI evaluates user performance based on tone, fluency, environments on their own. As a result, many talented confidences, and keyword relevance—offering insightful and individuals remain underprepared despite possessing the automated feedback through an interactive performance necessary knowledge and skills. dashboard.

The platform leverages React and Next.js for a modern, responsive frontend, Drizzle ORM and PostgreSQL for efficient backend data management, and scalable cloud services for media storage and AI integration. It is designed to be userfriendly, modular, and extensible, catering to students, earlycareer professionals, and even experienced job-seekers looking to refine their interview skills.

The objective of this paper is to present the motivation, architecture, features, and real-world use cases of the proposed system. It highlights the technical components used, the logic behind AI-driven feedback, and how such a platform can democratize access to high-quality interview preparation. By offering a self-paced, intelligent, and accessible solution, the system empowers users to bridge the gap between knowledge acquisition and interview performance—anytime, anywhere.

II. RELATED WORK

Mock interview platforms have gained considerable popularity in recent years as the demand for effective interview preparation has increased. Several platforms have emerged in this space, each offering its unique approach to simulating interview experiences. However, most existing solutions are constrained by fixed question banks, dependence on human interviewers, or limited adaptability.

Interviewing.io provides anonymous technical interview practice with real engineers from top tech companies. While effective, it largely depends on the availability of human interviewers and follows a one-size-fits-all approach to questioning. Similarly, Pramp enables peer-to-peer mock interviews, matching users with one another to alternate between the roles of interviewer and interviewee. Although collaborative, this format lacks consistency in evaluation and may not offer expert feedback.

HackerRank and LeetCode are among the most popular platforms for coding interviews, offering extensive problem sets and real-time coding environments. However, these platforms focus primarily on algorithmic problem-solving and do not replicate the behavioral or verbal aspects of interviews. Furthermore, they lack integrated systems for real-time video feedback or AI-driven performance analysis.

In contrast, our system leverages generative AI (Gemini) to create context-aware, personalized interview questions. By analysing the user's resume or selected job role, the system generates questions tailored to the individual's background and the target position. This approach not only improves relevance but also offers a more dynamic and engaging interview experience.

From a technical architecture perspective, many traditional systems adopt a monolithic backend structure, which can become a bottleneck as the platform scales. Our system employs a modern serverless architecture using Next.js Server Components and Edge Functions to optimize rendering and reduce latency, especially during AI inference and video handling. This microservices-inspired architecture enhances scalability, performance, and maintainability.

For user authentication and session management, platforms often use services like Firebase Authentication or Auth0. While powerful, these solutions can add complexity and overhead. We opt for Clerk, a dedicated authentication platform that simplifies integration with frontend frameworks like React and provides out-of-the-box support for multi-factor authentication, user management, and session handling. This ensures a smoother developer experience and a more secure authentication flow.

Other related innovations include AI tools such as HireVue and myInterview, which use pre-recorded video responses and AIbased evaluation. However, these are primarily employer-facing solutions designed for candidate screening rather than for candidate-side preparation. Our platform flips this perspective— empowering candidates to take control of their preparation with real-time AI feedback, a

performance dashboard, and repeatable practice sessions.

In summary, while existing platforms have addressed certain aspects of interview preparation, none have integrated real-time video interaction, generative question creation, AI feedback, and a modular full-stack architecture in a single platform. Our solution stands out by offering a holistic, interactive, and intelligent mock interview experience optimized for accessibility, personalization, and scalability.

III. SYSTEM ARCHITECTURE

A. Architecture Overview

The proposed AI-powered mock interview platform provides a comprehensive, real-time simulation of technical and behavioural interviews. This section elaborates on each step in the use case flow, defining the actors, preconditions, process flow, and outcomes for clarity.

A. User Registration/Login (via Clerk)

Actor: User (Student/Job Seeker/Professional)

Precondition: User visits the platform.

Flow:

1. The user selects “Sign Up” or “Login.”
2. Clerk handles authentication through email/password or OAuth (e.g., Google, GitHub).
3. Upon successful authentication, a user profile is either created or retrieved via Clerk.

Outcome: User is redirected to the personalized dashboard.

B. Resume Upload (Optional)

Actor: User

Precondition: User is authenticated and on the dashboard.

Flow:

1. The user clicks the “Upload Resume” option.
2. The system accepts a PDF or DOCX file.
3. A backend script or AI model parses the resume.
4. Extracted metadata (skills, experience, education) is stored in the PostgreSQL database via Drizzle ORM.

Outcome: Parsed resume data is used to generate personalized interview questions.

C. Select Job Role or Interview Type

Actor: User

Precondition: User is logged in and optionally uploaded a resume.

Flow:

1. The user selects the desired job role (e.g., Frontend Developer, Data Analyst).
2. The user chooses the interview type: Technical, Behavioral, or Mixed.
3. These parameters are stored in the session/context.

Outcome: System prepares for interview initialization with the selected parameters.

D. Start Interview: Questions Generated by Gemini AI

Actor: System (Gemini AI), User

Precondition: Job role and interview type are selected.

Flow:

1. The user clicks the “Start Interview” button.
2. The frontend triggers an API call to the backend service.
3. Gemini AI generates a contextual question using resume content, selected role, and interview type.
4. The question is displayed on the screen.

Outcome: Interview begins with dynamically generated questions.

E. Response Capture: Video Recorded via Webcam

Actor: User

Precondition: Interview session is in progress.

Flow:

1. The system activates the webcam and microphone using browser-based MediaStream APIs.
2. The user responds verbally to the displayed question.
3. The response is recorded and stored temporarily (either inmemory or a temporary storage bucket).

Outcome: Video response is saved and queued for analysis.

F. Feedback Generation (AI Evaluation)

Actor: System (Gemini AI, NLP Models)

Precondition: User response has been recorded.

Flow:

1. The system transcribes the video using a Speech-to-Text (STT) API.
2. The transcribed text is analyzed based on the following criteria:

○ Tone & Emotion: Evaluated via sentiment analysis models.
○ Keyword Relevance: Matched with expected job-specific keywords.

○ Fluency: Assessed using grammar and hesitation detection models.

3. The system calculates individual scores and generates qualitative and quantitative feedback.

Outcome: User receives AI-evaluated feedback for each question.

G. Performance Dashboard

Actor: User, System

Precondition: Interview session is complete.

Flow:

1. The system aggregates analytics across all responses.
2. Key performance metrics are displayed, including:

○ Confidence (based on tone and expression),

○ Fluency (grammar accuracy and hesitation rate),

○ Relevance (keyword and domain match).

3. Results are presented visually through graphs and interactive components.

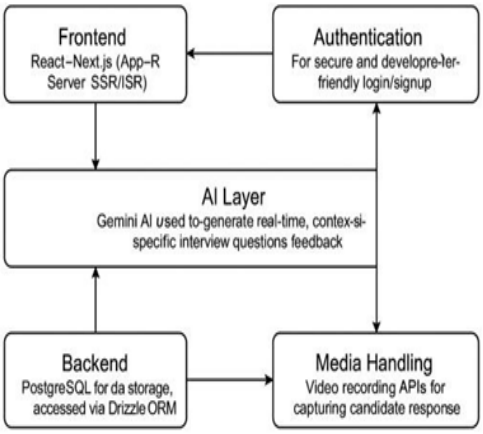


Figure 2: System Architecture Diagram

B. Use Case Flow

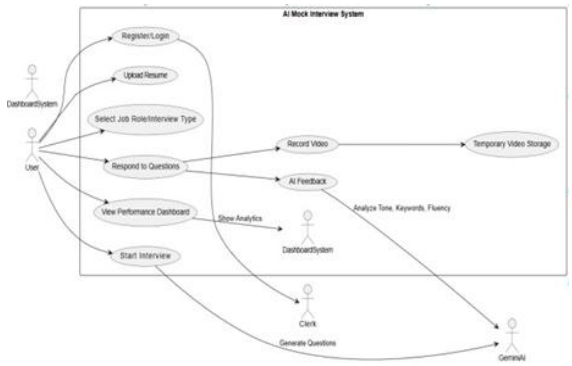


Figure 1: Use Case Diagram

IV. MODULES DESCRIPTION

Module	Description
Auth Module	Handles registration, sign-in, session tracking using Clerk
Interview Engine	Manages real-time communication with Gemini AI for question generation
Response Recorder	Integrates with browser media APIs to record and store responses
Feedback Engine	Uses AI/NLP techniques to evaluate answers and generate structured feedback
Analytics Dashboard	Visualizes user performance over time (charts, summaries, scores)

V. TECHNOLOGY STACK

Layer	Technology
Frontend	React, Next.js, Tailwind CSS
Backend	Next.js API Routes, PostgreSQL

Authentication	Clerk
ORM	Drizzle
AI/LLM	Gemini AI (Google)
Hosting	Vercel
Tools	Framer Motion, Lucide, shadcn/ui, Recharts

VI. AI QUESTION GENERATION

The core of the interview question generation process relies on Gemini AI's large language model (LLM), which is used to generate personalized and relevant interview questions. This section outlines how the system utilizes the candidate's resume, the desired role, and past performance history to generate tailored interview questions.

3.1 System Inputs

The system takes several inputs to personalize the question generation:

1. User's Resume: The candidate's resume serves as the primary input. It is parsed to extract key skills, technologies, and experiences. Techniques such as keyword extraction and named entity recognition (NER) are applied to identify crucial elements such as programming languages (e.g., Java, Python, JavaScript), tools (e.g., React, Docker, MySQL), and frameworks (e.g., Spring Boot, TensorFlow).

2. Desired Role: The specific role that the candidate is applying for (e.g., "Software Engineer Intern," "Data Scientist") is also provided as input. This contextualizes the questions to ensure they are relevant to the job being applied for.

3. Past Performance: The system stores and utilizes the candidate's previous interactions with the system, including the questions asked in prior interviews and their performance. This history helps avoid repetitive questions and ensures that the level of difficulty is appropriate to the user's expertise and previous responses.

3.2 Generating Questions with Gemini AI

Once the system has gathered all relevant inputs (resume, role, and past performance), it generates interview questions using the Gemini AI's LLM via an API call. The process follows these steps:

- Prompt Structure: The prompt given to Gemini AI is constructed to include the user's resume, the target role, and the specific technology or domain of interest. For example:

"Act as an interviewer for a Software Engineer intern position. Based on this resume, ask a medium-difficulty question about React."

This prompt ensures that the question generated is aligned with both the candidate's skillset (e.g., React) and the role (e.g., Software Engineer Intern). The AI model uses this information to generate a question that is specific to the candidate's profile.

Question Output: The output generated by Gemini AI is a natural language question. For example:

"Can you explain how React's Virtual DOM works and why it improves the performance of web applications?"

- This question is tailored to test the candidate's understanding of React, a technology mentioned in their resume.

3.3 Parsing and Storing the Output

The generated question is parsed by the system and stored in a

structured format for further processing. This allows the system to track all questions that have been asked to the user and helps prevent repetition in future interactions. The system also logs each generated question alongside the user's performance metrics (e.g., response time, correctness) to enhance future question generation.

3.4 Displaying the Question to the User

Once parsed and stored, the generated question is displayed to the user in a suitable format, either through a web interface or an interactive platform. The question is ready for the candidate to answer, initiating the next step in the interview process.

3.5 Advantages of the Approach

- Personalization: By considering the user's resume and desired role, the AI generates highly relevant and customized questions.

- Non-Repetitiveness: The system checks past performance to ensure that candidates are not asked the same question repeatedly, improving the efficiency of the interview process.

- Scalability: The system can generate a wide variety of questions for multiple users, supporting diverse roles and industries.

IV. EVALUATION AND TESTING

In order to assess the effectiveness and user satisfaction of the
AI-based interview question generation system, a comprehensive evaluation was conducted. This evaluation involved a closed-group testing phase with 25 beta testers, consisting of students and job seekers. In addition to user feedback, automated testing was also performed to ensure the reliability and stability of the system. This section details both the user feedback metrics and the automated testing procedures.

4.1 User Evaluation

The evaluation was designed to gather insights into the usability, relevance, and overall performance of the system. The 25 beta testers were asked to engage with the system in a series of simulated interviews and provide feedback based on various criteria. The key feedback metrics and their respective results are as follows:

Relevance of Questions:

○ Metric: 91% of users expressed satisfaction with the relevance of the questions generated by the system.

○ Details: Users reported that the AI-generated questions were closely aligned with their resume and the desired job role. The questions reflected an appropriate level of complexity based on their skill set, with many users commenting that they felt the questions were targeted specifically to their expertise and role.

2. Ease of Use:

○ Metric: The system received an average rating of 4.6/5 for ease of use.

○ Details: Testers found the system intuitive and userfriendly. The interface for interacting with the interview process, reviewing questions, and providing answers was straightforward. Most users mentioned that they had minimal difficultynavigating through the system and understood how to interact with the generated questions.

3. Feedback Usefulness:

○ Metric: 85% of users found the feedback provided by the system to be useful.

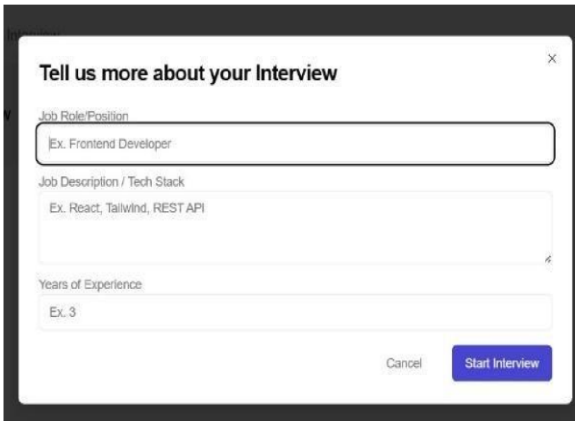
○ Details: The feedback provided to users after each response was seen as valuable. Users appreciated the insights into their performance, including areas where they excelled and areas for improvement. Many users highlighted that the feedback helped them understand what aspects of their responses were important in the context of the interview.

4. Overall Experience:

○ Metric: The system received an average rating of 4.5/5 for overall user experience.

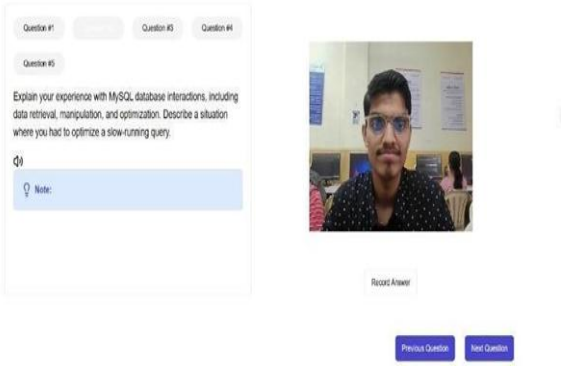
○ Details: The majority of beta testers reported a positive overall experience, citing that the system provided a realistic and insightful interview simulation. Testers enjoyed the personalized nature of the questions and the real-time feedback they received, which contributed to an enhanced interview preparation experience.

VI. RESULT



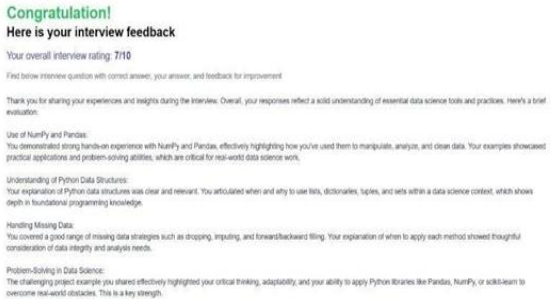
AI-Generated Interview Feedback Summary

This screen displays a detailed performance summary post interview, with an overall rating and feedback across different areas like NumPy, Pandas, Python data structures, and handling missing data. It provides personalized, structured insights for improvement in data science interview skills.



Live Interview Interface with Webcam

The interface shows a real-time interview question and a webcam preview for recording the response. Users interact with AI-generated queries, such as explaining MySQL optimization strategies, while the system captures responses for analysis.



VI. CONCLUSION

In conclusion, this paper presented the development of an intelligent, full-stack mock interview platform that redefines how job seekers and students prepare for technical interviews. By integrating advanced tools such as Next.js, Clerk, Drizzle ORM, and Gemini AI, the platform delivers a secure, scalable, and dynamic experience tailored to individual user needs. The system's ability to generate domain-specific questions, analyze webcam-recorded responses, and provide real-time, actionable feedback makes it a powerful resource for continuous improvement. With features like progress tracking and personalized analytics, users are empowered to refine their skills and gain the confidence required to excel in real-world interviews. Future enhancements, including expanded job role support and non-verbal cue analysis, position this platform as a comprehensive solution in the evolving landscape of AI-driven career preparation.

Keywords: AI Mock Interview, Full-Stack Development,
Next.js, Gemini AI, Drizzle ORM, Clerk, Career Readiness,
Real-Time Feedback, Interview Simulation

REFERENCES

[1] Google. "Gemini AI." [Online]. Available: https://deepmind.google/technologies/gemini
[2] Clerk.dev. "Authentication for Modern Apps." [Online]. Available: https://clerk.dev
[3] Drizzle ORM Documentation. [Online]. Available: https://orm.drizzle.team
[4] Next.js Framework. [Online]. Available: https://nextjs.org
[5] [5] PostgreSQL Documentation. [Online]. Available: https://www.postgresql.org
[6] Framer Motion Animation Docs. [Online]. Available: https://www.framer.com/motion/
[7] OpenAI, "GPT-3: Language Models are Few-Shot Learners," *arXiv preprint arXiv:2005.14165*, 2020. [Online]. Available: https://arxiv.org/abs/2005.14165
[8] Amazon Web Services (AWS), "AWS Cloud Services," [Online]. Available: https://aws.amazon.com
[9] ReactJS, "React: A JavaScript library for building user interfaces," [Online]. Available: https://reactjs.org
[10] TensorFlow, "TensorFlow: An Open-Source Machine Learning Framework," [Online]. Available: https://www.tensorflow.org
[11] Docker, "Docker: Empowering developers to simplify containerization," [Online]. Available: https://www.docker.com
[12] Microsoft, "Azure AI," [Online]. Available: https://azure.microsoft.com/en-us/services/cognitive-services/
[13] [13] GitHub, "GitHub: Where the world builds software," [Online]. Available: https://github.com
[14] IBM, "Watson AI," [Online]. Available: https://www.ibm.com/watson

[15] OpenCV, "OpenCV: Open-Source Computer Vision," [Online]. Available: https://opencv.org