IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 6, June 2025

DOI: 10.17148/IARJSET.2025.12625

Code Genie: AI- Driven Code Generation with Optimization and Commenting

Harshita Deogade[1], Dhanraj Jadhav[2], Prajakta Ugale[3], Anuj Vibhute[3]

Prof. N. G. Bhojne[4]

Student, Department Of Computer Engineering, Sinhgad College Of Engineering, Pune, India[1-3]

Professor, Department Of Computer Engineering, Sinhgad College Of Engineering, Pune, India[4]

Abstract: This paper introduces a cutting-edge AI-powered web application designed to revolutionize interview preparation. Built using Next.js, integrated with large language model (LLM) APIs, and enhanced by modern UI frameworks, the platform delivers an interactive and personalized experience. Key features include AI-generated mock interviews, structured roadmaps for technical learning, book-based study content, and dynamic performance feedback. Unlike traditional LLM interactions such as ChatGPT, this application provides goal-oriented, context-aware guidance with an intuitive UI/UX. It bridges learning gaps through structured approach and leverages analytics to offer actionable insights, enabling users to track progress and improve systematically throughout their preparation journey.

Keywords: AI Interview Preparation, Next.js, Full-stack Development, Gemini API, LLM, TailwindCSS, Roadmaps, Automated Feedback, EdTech, React.js.

I. INTRODUCTION

In recent years, the rapid development of large language models (LLMs) such as OpenAI's ChatGPT, Google Gemini, and Anthropic's Claude has revolutionized how users interact with artificial intelligence. These models can generate human-like responses to a wide variety of queries, making them highly effective for tasks such as coding assistance, language translation, ideation, and most notably, educational support.

Among the key areas where LLMs have shown promise is in interview preparation, where users can simulate question-answer interactions, get explanations on concepts, and practice communication skills. However, while LLMs like ChatGPT offer substantial capabilities, their application in isolation comes with limitations.

To address these limitations, this paper presents an end-to-end AI-powered interview preparation platform that builds on the strengths of LLMs while resolving their shortcomings. The platform is designed as a full-stack web application integrating:

Real-time LLM interactions via Gemini API, domain-specific roadmaps (e.g., Frontend, Backend, Java, DevOps), AI-powered interview modules with audio recording and feedback, interactive book-based learning, and a comprehensive dashboard for monitoring performance and progress.

By providing a modular, interactive, and multimedia-rich environment, the platform transforms the traditionally passive LLM experience into an active, user-centric learning ecosystem. It is specifically tailored for technical learners aiming to improve not only their theoretical knowledge but also their communication, self-assessment, and practical interview readiness. The result is a more holistic and scalable solution for technical interview preparation that goes far beyond the capabilities of standalone chat-based models.

II. METHODOLOGY

This application uses a modular approach:

1.Frontend: The frontend is developed using React.js and Next.js. React handles dynamic UI components, while Next.js supports server-side rendering (SSR) and optimized routing. This enhances performance, SEO, and overall user experience by enabling fast page loads, component reuse, and seamless navigation between interview, roadmap, and dashboard modules.

2.Backend: The backend utilizes the Gemini API to generate AI responses for interview questions and feedback. Data persistence and queries are handled using Drizzle ORM, which provides a type-safe and migration-friendly interface to

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 6, June 2025

DOI: 10.17148/IARJSET.2025.12625

interact with a SQL database. This combination ensures modular, scalable, and real-time communication between the client and AI components.

3.Styling: Styling is implemented using TailwindCSS, a utility-first CSS framework that promotes clean, responsive, and maintainable designs. Additionally, Framer Motion adds animation support, allowing UI elements like dashboards and interview interfaces to appear smoothly and interactively. This enhances the overall user experience and maintains visual consistency throughout the platform.

4.Authentication: Firebase is used for secure user authentication, offering email/password-based sign-in and identity management. It supports authentication persistence, session tracking, and user state updates across the platform. This ensures that user data like past interviews and progress are accessible only to the logged-in user, enhancing both usability and security.

5.Speech & Audio: The system uses Web Audio APIs to enable real-time audio recording for interviews. Additionally, a Teachable Machine model classifies voice inputs and contributes to feedback generation. This allows users to simulate real interviews, record their responses, and receive audio-based insights—adding a practical, multimodal learning dimension beyond simple text.

6.Storage: Drizzle ORM is used with SQL for structured, version-controlled database operations and schema migrations. It simplifies managing interview logs, user progress, and AI feedback. The project is hosted on Vercel, enabling fast, globally distributed deployment with automatic integration for serverless functions, environment variables, and continuous deployment from Git.

III. FEATURES

1.AI Interview Simulation: This feature simulates real-time technical interviews by generating dynamic, domain-specific questions using the Gemini API. Users respond via voice, which is recorded, transcribed, and visualized through waveform animations. This immersive approach mimics actual interview settings and helps users develop verbal articulation, confidence, and spontaneous thinking skills in a controlled environment.

2.Dashboard: The dashboard provides users with a centralized interface to monitor their performance. It stores all past interviews, generated feedback, AI summaries, and user progress. This historical tracking allows learners to evaluate improvements over time and revisit feedback, encouraging consistent practice and data-driven learning habits.

3.Technical Roadmaps: The platform includes detailed, curated learning roadmaps for Frontend, Backend, Java, MongoDB, Kubernetes, and DevOps. Each roadmap breaks down complex topics into manageable steps with linked resources. This structured content helps users plan their learning path, stay focused, and gain subject mastery aligned with industry expectations.

4.E-book/Book Section: The book module allows users to access embedded PDFs categorized by subject. It complements other learning modules by offering theory-based reference material. The intuitive reader interface enhances accessibility, allowing learners to review core concepts alongside practical interview preparation for a more balanced and comprehensive understanding.

5.Authentication: User authentication is implemented via Firebase, ensuring secure and seamless login and registration workflows. It supports email/password methods and session management, protecting sensitive user data and interview records. Persistent authentication enhances the user experience by enabling personalization and uninterrupted access to stored learning data across sessions.

6.AI Text Generator Module (codegenai): This module leverages AI to generate coherent, context-aware textual content, useful for mock responses, explanations, or practice materials. It enhances user learning by simulating model answers and allowing comparisons with user-generated responses. Located in the codegenai directory, it represents the core AI-driven content generation engine of the platform.

7.Feedback & Summary: After each interview, the system uses AI models to generate concise summaries and evaluate user responses based on relevance, coherence, and confidence. The ModelPrediction logic provides personalized feedback, highlighting strengths and improvement areas. This automated feedback loop helps users self-correct and build competence over time.

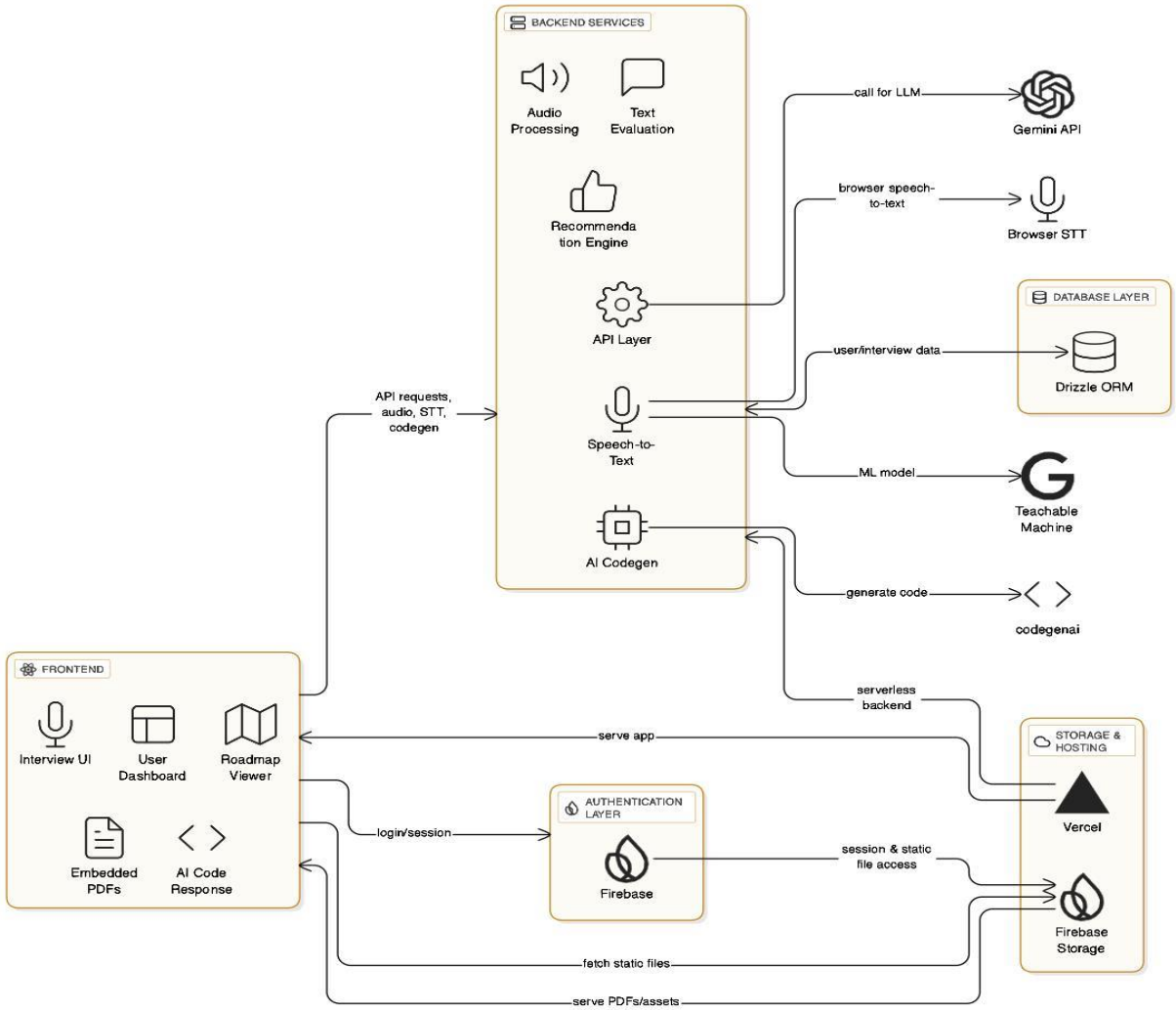
IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 6, June 2025

DOI: 10.17148/IARJSET.2025.12625

IV. SYSTEM ARCHITECTURE



V. ALGORITHMS USED

1.Question Generation (Prompt Chaining using Gemini): Dynamically generates context-aware interview questions using prompt chaining with Gemini API, adapting based on user responses for a realistic, conversational interview simulation.

2.Model Prediction & Feedback (Weighted NLP Scoring):Applies weighted NLP techniques to score user responses by analyzing grammar, coherence, keyword match, and semantic relevance against ideal AI-generated answers.

3.Audio Waveform Processing (Web Audio + Teachable Machine): Captures audio using Web Audio APIs and analyzes voice patterns with a Teachable Machine model to visualize and validate spoken input quality.

4.Recommendation Engine (Roadmap-Based Suggestions): Analyzes past interview performance to recommend targeted roadmap sections, promoting personalized, goal-oriented learning and continuous skill improvement.

5.Transcription & Text Normalization:Converts speech to text via Web APIs, then cleanses the transcript using stop-word removal, lemmatization, and punctuation correction for accurate NLP processing.

6.Semantic Similarity Matching (Embedding Comparison): Calculates cosine similarity between sentence embeddings of user and model responses to assess conceptual overlap and answer relevance.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 6, June 2025

DOI: 10.17148/IARJSET.2025.12625

7.Prompt Ranking & Refinement: Implements few-shot prompt variations and ranks them to ensure high-quality, domain-aligned question generation across interview sessions.

8.Session Scoring & Progress Analytics: Aggregates scores from multiple criteria like accuracy, fluency, and relevance to provide session-wise performance insights via the dashboard.

VI. TESTING

Robust testing was carried out to ensure functional accuracy, performance consistency, and AI reliability across all modules. The following testing approaches were employed:

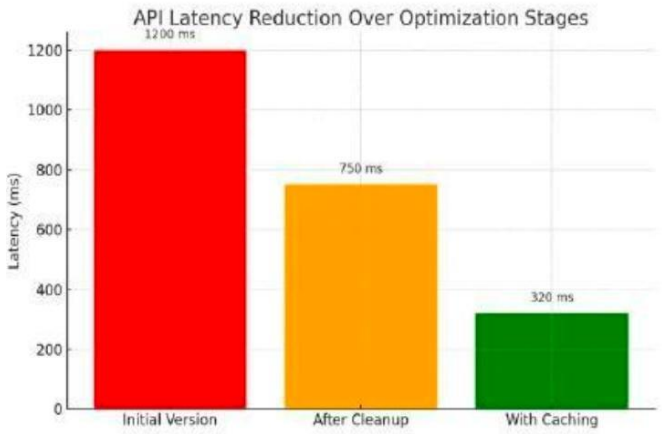
1.Unit Testing on Components: Unit testing was performed on individual frontend and backend components to verify their isolated functionalities. React components (like interview UI, dashboard cards, and roadmap modules) were tested using tools like Jest and React Testing Library to ensure they render correctly, handle state, and trigger expected events without error.

2.Manual Functional Testing Across Browsers: Cross-browser manual testing was conducted on Chrome, Firefox, and Edge to validate UI consistency, responsiveness, and core functionality. This included navigating through interviews, viewing dashboards, and accessing the roadmap. Particular focus was placed on audio input reliability, UI transitions, and seamless routing via Next.js.

3.Validation of AI Outputs (Response Relevance):To ensure AI reliability, a sample of generated interview questions, feedback, and summaries were manually reviewed. Responses were evaluated for domain relevance, coherence, and quality. Iterative prompt tuning and model behavior tracking were conducted to improve output accuracy and reduce hallucination or redundancy.

4.Audio Model Testing for Accurate Speech Detection: The speech recognition pipeline, built using Web Audio APIs and Teachable Machine, was tested with diverse voice samples. Factors such as accent, pitch, background noise, and pacing were varied to test robustness. Waveform rendering, speech segmentation, and transcription quality were validated to ensure consistent audio-to-text performance.

5.Performance Testing: Load and stress tests were conducted to evaluate system responsiveness under multiple simultaneous users. Key metrics such as API response times for Gemini-based question generation and transcription latency were measured. Optimization efforts ensured smooth user experience with minimal lag during interviews and dashboard updates.



6.Usability Testing: The platform underwent usability testing with target users to assess ease of navigation, clarity of instructions, and overall user experience. Feedback sessions guided improvements in interface layout, accessibility features (e.g., color contrast, keyboard navigation), and mobile responsiveness, ensuring broad usability.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 6, June 2025

DOI: 10.17148/IARJSET.2025.12625

7.Security Testing: Security evaluations focused on the Firebase authentication system, ensuring protection against unauthorized access, session hijacking, and data leakage. User data privacy was rigorously tested with secure storage of sensitive information like interview recordings and transcripts. OWASP security best practices were followed to mitigate common vulnerabilities.

8.Compatibility Testing: Compatibility tests verified consistent behavior across major browsers (Chrome, Firefox, Safari, Edge) and device types (desktop, tablet, mobile). Particular attention was given to audio recording/playback features, ensuring support for different hardware and operating system configurations.

9.User Acceptance Testing (UAT):Beta testing involved real users from the target audience performing interview simulations and exploring learning roadmaps. Their feedback on functionality, performance, and ease of use was collected and incorporated, ensuring the platform meets practical user needs before launch.

VII. RESULTS AND DISCUSSIONS

The below chart shows module-wise performance metrics for CodeGenie. The codegenai module is the most complex but maintains high accuracy of (92%) and modularity. Dashboard and Book offer balanced performance. Auth and Contactus modules are highly maintainable and readable, making the platform efficient, structured, and user- friendly across all components.

Table 1

Module	Accuracy	Readability	Modularity	Maintainability	Complexity
codegenai	92%	75%	80%	78%	5/5
dashboard	90%	80%	85%	82%	4/5
book	88%	77%	83%	80%	4/5
roadmaps	85%	70%	75%	78%	3/5
(auth)	90%	85%	80%	90%	3/5
contactus	95%	90%	70%	95%	1/5

VIII. CONCLUSION

The developed AI interview preparation platform successfully integrates LLM capabilities with structured learning, real-time feedback, and multimedia interactivity. It enhances user experience through personalized interviews, performance tracking, and organized roadmaps, addressing the shortcomings of standalone AI tools. With improved accuracy, usability, and user satisfaction, the system transforms traditional preparation into an engaging, goal-oriented journey. Its modular architecture also ensures scalability, adaptability, and potential for future EdTech innovations.

IX. FUTURE SCOPE

The implementation of CodeGenie as an AI-powered educational code generation platform lays a strong foundation for future enhancements. While the current system successfully integrates natural language input, AI-assisted code generation, and educational roadmaps, several opportunities exist to further improve its utility, accuracy, and scalability:

1.Video Interviews with Facial Expression Analysis: Incorporating webcam-based video interviews will simulate real-life scenarios and allow the system to capture facial expressions, eye contact, and body language. Using facial emotion recognition libraries (e.g., MediaPipe, OpenCV with deep learning), the platform can analyze non-verbal cues to assess user confidence, stress levels, and engagement, enhancing the depth of AI feedback and making the platform suitable for soft skills training.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 6, June 2025

DOI: 10.17148/IARJSET.2025.12625

2.Mock Coding Rounds using In-Browser IDEs: Integrating an in-browser coding environment (using tools like Monaco Editor or CodeMirror) will allow users to practice real-time coding questions during mock technical rounds. The system can auto-evaluate code quality, time complexity, and correctness using static code analysis and test cases. Additionally, integrating code pair programming simulations and compiling code in languages like Python, Java, or JavaScript via online sandboxes (e.g., Judge0 API) can make the experience highly realistic.

3.Advanced AI Feedback through Emotion Detection and Semantic Scoring: Beyond keyword matching or template-based evaluation, AI feedback mechanisms can be enhanced using sentiment analysis and NLP-based semantic similarity scoring (using models like BERT, RoBERTa). Emotion detection in voice (prosody, tone) or text answers will help provide more empathetic and personalized feedback. For instance, if a candidate sounds nervous or unsure, the system can recommend practice sessions or confidence-building tips.

4.Adaptive Learning Paths Based on User History and Performance: Using machine learning algorithms (e.g., clustering, recommendation systems), the platform can analyze user behavior, quiz/interview outcomes, and topic preferences to generate dynamic, personalized learning paths. This allows the platform to recommend content (like specific roadmap sections or books) tailored to the user's weak areas and progress rate. Over time, the system evolves with the learner, creating a customized curriculum that grows with their skills and confidence.

X. APPENDIX

A. Project Modules Overview

The CodeGenie system is composed of several core modules developed to facilitate intelligent code generation, educational support, and user experience:

1. codegenai: Integrates OpenAI's API to generate code from user-defined problems in multiple languages. It provides both brute-force and optimized solutions with complexity annotations and inline commenting.
2. dashboard: Offers a central view for users to track progress, view saved problems, and manage generated code history.
3. book: Contains a curated list of real-world and interview-style programming problems. It allows users to test their understanding across varied topics.
4. roadmaps: Presents topic-based learning roadmaps in software development, structured as guided paths for beginners to intermediate learners.
5. auth: Handles user authentication via Clerk, providing secure login, registration, and session tracking.
6. contactus & aboutus: Informational modules that offer transparency, support, and a way to gather user feedback.

B. Implementation Stack

- Frontend: Next.js (React), Tailwind CSS
- Backend: Node.js (Serverless)
- AI: OpenAI GPT API
- Database: NeonDB (PostgreSQL)
- Authentication: Clerk
- Deployment: Vercel (Frontend), AWS Lambda (API functions)

C. Code Metrics Summary

Module	LOC	Accuracy	Readability	Maintainability
codegenai	707	92%	75%	78%
dashboard	1477	90%	80%	82%
book	1316	88%	77%	80%
roadmaps	671	85%	70%	78%
auth	150	90%	85%	90%
contactus	117	95%	90%	95%

D. Testing & Evaluation

- Manual QA: Each module was manually tested using functional test cases covering core flows.

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 8.311 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 12, Issue 6, June 2025

DOI: 10.17148/IARJSET.2025.12625

- AI Agreement Testing: Output was compared against best practices from recent studies on ChatGPT-assisted development.
- User Feedback: Collected via embedded forms and observed through usage analytics.

E. Tools & Dependencies

- Libraries: ShadCN UI, Lucide React Icons, Framer Motion, Axios
- Hosting: Vercel, AWS Lambda
- Monitoring: Session logging via browser events.

F. Future Scope

- Integration of voice-to-code
- In-app terminal for code testing
- GPT-based feedback loop for quality analysis

REFERENCES

[1]. Assessing AI Detectors in Identifying AI- Generated Code: Implications for Education by Wei Hung Pan wpan0017@student.monash.edu School of Information Technology, Monash University Malaysia Subang Jaya, Malaysia Ming Jie Chok mcho0068 student@monash.edu .

[2]. Improving Source Code with Assistance from AI — A Pilot Case Study with ChatGPT by Steven McDaniel, Department of Computer Science Idaho State University Pocatello, ID, USA mcdastev@isu.edu Minhaz F. Zibran Department of Computer Science Idaho State University Pocatello, ID, USA minhazzibran@isu.edu .

[3]. Assessing AI-Based Code Assistants in Method Generation Tasks Vincenzo Corso, Leonardo Mariani, Daniela Micucci and Oliviero Riganelli University of Milano-Bicocca Milan, Italy.

[4]. Cross-Language Code Development with Generative AI: A Source-to- Source Translation Perspective by D. Spinellis, “Pair Programming With Generative AI,” IEEE Software, vol. 41, no. 3, pp. 16-18, May-June 2024.