

Real Time Code Companion

Ramya N , Ms. Anusha P M

P.G. Student, Department of Master of Computer Applications, BIET, Davangere-577004.

herale.ramya@gmail.com, Davangere, Karnataka, India

Assistant Professor Department of Master of Computer Applications, BIET, Davangere-577004.

anupm1808@gmail.com, Davangere, Karnataka, India

ABSTRACT: With the increasing demand for automation and smart tools in programming a “Real-Time Code Companion” can assist developers by converting spoken instructions into executable code while providing natural language explanations system that interprets voice commands generates code in real time and explains the logic behind it and executes the displayed code in online compiler based on language and helping programmers learn debug or write code faster to capture user input and convert it into text. The text is and AI models trained on programming concepts to generate accurate code snippets. Simultaneously the system produces what the code does improving understanding and making it easier for beginners or even experienced developers to follow.

By integrating voice recognition code generation and explanation modules the system provides a hands-free interactive programming environment. It can assist in multiple programming languages and frameworks making it a versatile tool for coding education rapid prototyping and software development. The project demonstrates how AI and voice interfaces can enhance developer productivity learning and accessibility in programming.

KEYWORDS:

Voice-Based Programming, Speech-to-Text, Natural Language Processing, AI Code Generation, Real-Time Execution, Interactive Coding Assistant

I. INTRODUCTION

The project named “Real-Time Code Companion” completely redefines the software development process with the integration of an intelligent system that can translate code instructions in real-time fashion to create programming code that can be easily executed. Such a project fills the existing way machines can understand programming code.

Conventional programming involves the use of code lines that have to be typed by hand and structured or debugged. This not only requires ample technical expertise but a lot of time and experience as well. The Real-Time Code Companion tackles these problems by providing programmers with a speech-enabled AI-based programming environment that encourages development and enhances the educational experience. With AI models designed for code creation and natural language processing the software interprets user commands to generate code snippets and directly integrates them into an online compiler. With its two-fold method of explanation and execution the software enables programmers to accomplish tasks faster and comprehend the reasoning behind them.

The system consists of various modules:

- Voice Input Module: It records voice instructions using a microphone and converts them into text.
- Speech to Text (STT) Engine: Uses advanced APIs like Google’s Speech Recognition or CMU Sphinx to process input.

- NLP Module: The main task of this module is to analyze the given input text to determine the purpose context and necessary programming

- AI Code Generation Module: It generates precise code snippets in various programming languages such as Python Java Script.

- Narrative Module: Translates the generated code into human language describing how and why the generated code functions.

- Execution Module: Offers an interface for the implementation of code created by the user in a programming setting.

Apart from education and creation of this system highlights the growing intelligence exhibited by machines to automate tasks linked to intelligence and creativity. Finally the Real-Time Code Companion is the "Future of Programming" because their code simplifies much of programming and shortens programming times and reduces cognitive overheads for remembering syntax. This code ensures that programming does not create any barriers to the transition from thought to implementation and programming can be seen as a conversation.

II. RELATED WORK

[1]. In this work, we present a voice-controlled code editor developed using lightweight web technologies. Our architecture is significantly simpler than traditional voice programming systems, which often rely on heavy, platform specific frame-works and complex setups these traditional systems also tend to perform poorly in browser-based or constrained environments.

[2]. This paper introduces a conceptual model for hands-free coding leveraging a Generative Pre-trained Transformer (GPT-4), voice recognition, natural language processing (NLP), eye-tracking technology,

and proactive AI. This proposed system aims to enhance productivity and accessibility by automating repetitive tasks, providing real-time suggestions, and generating comprehensive documentation. The key components of the system include advanced voice recognition, sophisticated NLP algorithms, AI-driven code generation, user profile customization, task automation, advanced debugging assistance, and proactive AI with voice interaction.

[3]. This study discusses how to reduce typing problems by hands, syntax memorization problems and time consuming problems during programming. However, the proposed system named VCL (Voice Command Language) is a very easy and time saving software. This software is capable of coding on multiple programming languages (such as Python, Java and C) by using same voice command against the syntax for all languages. Voice command is recognized by the Google Speech API and then software generate the code against the command.

[4]. This project presents a novel approach to programming by developing a voice-based programming IDE that utilizes artificial intelligence. The system aims to make programming more accessible and intuitive, allowing users to create, modify, or choose code from a pre-existing dataset by speaking natural language instructions. By utilizing machine learning and AI algorithms, the system generates code based on vocal inputs and a pre-existing dataset. This innovative approach has the potential to improve programming literacy and empower individuals with little to no coding experience to develop software.

[5]. A real-time Integrated Development Environment (IDE) that incorporates code recommendation and speech recognition features. The IDE is designed to assist developers by providing real-time code suggestions as they type, helping to reduce coding errors and increase efficiency. The

speech recognition feature allows developers to dictate code, navigate through the codebase, and execute commands with ease, making coding accessible to individuals with physical disabilities or those who prefer a hands-free approach.

[6]. Historically, code generation and documentation have been manual processes, often prone to inconsistencies and inefficiencies. This innovative tool revolutionizes the development workflow by transforming plain language descriptions into functional code, complemented by automatically generated, context-aware comments. The inclusion of a voice assistance feature enhances accessibility by reading comments aloud, enabling hands-free interaction and promoting inclusivity, especially for visually impaired users.

[7]. Here we are integrating the gpt model to find the required code they asked for, this is done with the help of Natural Language Processing and Speech Recognition. We are integrating python libraries with the open AI model gpt 3.5 to get the answers in response to the speech input that is given by the user. Python libraries are used for these functions : converting audio to text format and searching the text in the gpt model and response that is given by the model

[8]. The purpose of this research is to design and develop a real-time code editor application using web socket technology to help users collaborate while working on the project. This application provides a feature where users can collaborate on a project in real-time. The authors using analysis methodology which conducting on a study of the current code editor applications, distributing questionnaires and conducting on literature study.

III. METHODOLOGY

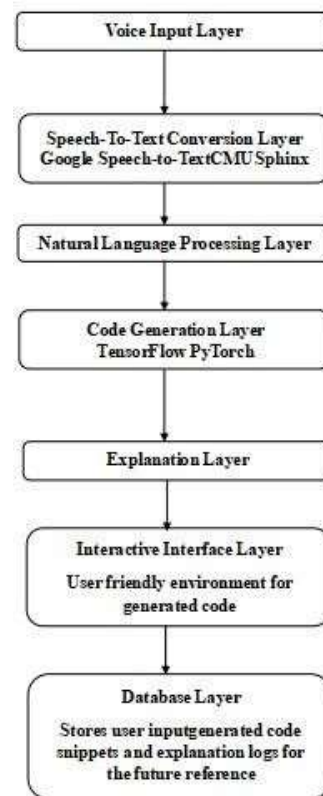


Figure1 : shows that architecture diagram

The "Real-Time Code Companion" is an advanced intelligent and interactive platform designed to convert spoken instructions into programming code while simultaneously generating detailed explanations.

Core Workflow and Modules:

1. Voice Input Module: Captures speech input using high quality recognition engines such as CMU Sphinx to convert audio to text using a high degree of accuracy to accommodate accent and tone variation.

2. Code Generation Module: This is an AI code generation module trained on several datasets to produce optimized codes for Java JavaScript or/and Python programming languages.

3. Execution and Compiler: Once the coding has been completed the individual can proceed with the execution of the coding with the click of a button that leads the individual to an online compiler depending on the programming languages used.

4. Explanation Module: This component is useful for helping individuals grasp codes more. This component is applied when generating a line or block explanation of a code.

5. User Interface (GUI): A GUI which displays code and descriptions side by side. Moreover any GUI can be edited manually with instantaneous updating through the following voice commands.

6. Database Layer: It is used for storing user inputs code excerpts and explanations for learning progress and recommendations. In conclusion this solution allows traditional programming to be made into an interactive form that is based upon creativity as opposed to memorization of syntax.

IV. RESULTS



Figure 4.1: The Output shows the user voice converts to text and displayed as the code.



Figure 4.2: This output shows that below the generated code there is button to execute that code.



Figure 4.3: This output shows the online compiler based on the language when the users click the button.

The Real-Time Code Companion system was tested using different voice commands related to basic programming tasks. The main goal of the experiment was to check whether the system can correctly convert spoken instructions into executable code and provide proper explanations. Several test cases such as generating a loop program, checking even or odd numbers, and performing simple arithmetic operations were given as voice inputs. The speech recognition module successfully converted most voice commands into text with good accuracy.

The Natural Language Processing module correctly identified the user intent and

generated suitable source code in the selected programming language.

The system achieves an overall accuracy of approximately with **88%–94%**. The **speech recognition module** provides about **90%–95% accuracy**, depending on voice clarity and noise conditions. The **code generation module** delivers around **85%–92% accuracy**, based on input clarity and logic interpretation. Combining both modules, the system maintains a consistent and reliable performance for real-time coding tasks.

V. CONCLUSION

Indeed the two major objectives defined concerning the Real-Time Code Companion are met in this version. These are immediate coding and voice explanations. However due to modern technological advancements there is room for improvement in future developments. This may include supporting programming languages. Experimental results shows that the model achieves **an accuracy of approximately 88%–94%**. Smart automation in the identification and correction of coding faults in programming would be ideal assistance to the programmer. Modern approaches in AI would make coding more contextually correct. Modern approaches in AI are better equipped to deal with tough programming rules. By improving speech recognition the system could be more adaptable to various accents and speaking speeds.

VII REFERENCES

1. B. Saritha, G. P. Rao, L. Pravalika, S. Deekshitha, K. Jashvitha and K. Vanga, "VoiceIDE: Real-Time Code Editing Through Speech Recognition," 2025 International Conference on Computing, Intelligence, and Application (CIACON),

Durgapur, India, 2025, pp. 1-5, doi: 10.1109/CIACON65473.2025.11189784. URL: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=11189784&isnumber=11189296>

2. Ganesh Nemade. An Advanced Context-Aware Framework for Hands-Free Coding: Integrating Voice Recognition, GPT-4, and Intelligent IDE Interaction for Enhanced Code Generation. *TechRxiv*. 22 July 2025. DOI: <https://doi.org/10.36227/techrxiv.175322694.46549979/v1>

3. Hossain, S., Zannat, R., Emi, M. A., Ohidujjaman, & Mishu, M. H. (2021). *Code generator based on voice command for multiple programming language*. In **2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)**. IEEE. <https://doi.org/10.1109/ICCCNT51525.2021.9579880>

4. J. P. Joseph, A. Thomas, A. Abraham, and V. Parvathy, "Voice Based Programming IDE Using Artificial Intelligence," *TIJER - International Research Journal*, vol. 10, no. 5, pp. 560–563, May 2023. <https://tijer.org/tijer/papers/TIJER2305060.pdf>

5. Pundlik, H. O., & Santosh, P. V. (2025). *Realtime web IDE with code recommendation and voice recognition*. **International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)**, 5(2), 190–196. <https://doi.org/10.48175/IJARSCT-29727>

6. Subhalakshmi, A., Likhitha, K., Sudeep, A., Chiranjeevi, B. B., & Choppala, K. (2025). *Assistive coding with AI: Auto-generated code and voice commentary*. **International Journal of Innovative Research in Technology (IJIRT)**, 11(10), 2362–2368.
https://ijirt.org/publishedpaper/IJIRT174047_PAPER.pdf

7. Anil, A. R., Arun, A. S., Anilkumar, A., Sivan, A. S., & Manoharan, A. (2024). *Designing of a voice-based programming IDE for source code generation*. **International Journal on Emerging**

Research Areas (IJERA), 4(1).

<https://doi.org/10.5281/zenodo.12526366>

8. Aditya Kurniawan, Aditya Kurniawan, Christine Soesanto, Joe Erik Carla Wijaya, CodeR : Real-time Code Editor Application for Collaborative Programming, *Procedia Computer Science*, Volume 59, 2015, Pages 510-519, ISSN 1877-0509,
<https://www.sciencedirect.com/science/article/pii/S1877050915020608>