IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, March 2026
Impact Factor: 8.2

CodeLite: Smart Multi-Language Mobile Code Editor

Prof. Ashwini Wakodikar[1], Chetana Choudhary[2], Ranjana Tiwari[3]
Assistant Professor, Department of Computer Application[1]
PG Scholar, Department of Computer Application[2,3]
K.D.K. College of Engineering, Nagpur, Maharashtra, India
ashwini.wakodikar@kdkce.edu.in , chetanarchoudhary.mca24f@kdkce.edu.in
ranjanartiwari.mca24f@kdkce.edu.in

Abstract: *In the current digital learning environment, students and developers increasingly require flexible programming tools that support on-the-go coding and experimentation. Conventional desktop-based development environments demand complex installations, high system resources, and fixed workstations, which limit accessibility and convenience. This paper presents CodeLite, a smart mobile-based multi-language code editor designed to enable program writing and execution directly on Android devices through cloud-assisted compilation. The system processes user-written code using remote execution services, eliminating the need for local compiler setup. Codelite supports multiple programming languages and integrates secure user management and persistent storage to enhance usability and continuity. The application is built using a scalable mobile architecture combined with cloud services to ensure responsiveness and reliability. Experimental evaluation indicates that the system delivers accurate execution results with minimal latency, demonstrating its effectiveness as a portable programming and learning support platform.*

Keywords: Mobile Code Editor, Cloud-Based Compilation, Multi-Language Programming, Android Application, Remote Code Execution, Software Development Tools

I. INTRODUCTION

Programming has become a fundamental skill in modern education and software development, with increasing demand for flexible and accessible coding environments. However, traditional programming practices largely depend on desktop-based integrated development environments that require complex installations, high system resources, and fixed working setups. These constraints limit accessibility for students and developers who wish to practice coding outside laboratories or while on the move. As a result, there is a growing need for portable programming platforms that support real-time code development and execution.

The widespread adoption of smartphones and advancements in mobile computing have opened new possibilities for mobile-based programming tools. Despite this progress, most existing mobile code editors provide limited functionality, often restricted to basic text editing or syntax highlighting without actual program execution. Such limitations reduce their effectiveness for practical learning and real-world programming tasks. To address these challenges, cloud-based compilation has emerged as a promising approach by enabling remote execution of programs without relying on local compilers.

Cloud-assisted programming environments allow users to submit source code to remote servers where compilation and execution are performed, and results are returned instantly. This approach reduces device-side processing requirements while ensuring consistent execution across different platforms. When combined with mobile applications, cloud-based execution enables lightweight yet powerful programming solutions suitable for learning, experimentation, and rapid testing.

DOI: 10.48175/IJARSCT-31418

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, March 2026
Impact Factor: 8.2

II. LITERATURE REVIEW AND MOTIVATION

Several studies have examined the development of mobile-based programming tools and cloud-assisted execution platforms aimed at improving accessibility to coding environments. Early mobile code editors primarily focused on text editing and syntax highlighting, offering limited support for actual program compilation and execution. Such approaches restricted their usefulness for practical programming tasks and learning-oriented applications.

Recent research emphasizes the role of cloud-based compilation services in overcoming hardware and configuration constraints associated with traditional development environments. By enabling remote execution of source code, cloud-assisted systems eliminate the need for local compiler installation and ensure consistent execution across devices. Studies also indicate that integrating multi-language execution within a single platform enhances learning continuity and reduces dependency on multiple tools. However, many existing solutions are browser-dependent or lack proper optimization for mobile interaction, leading to performance and usability challenges.

Existing mobile programming platforms often focus on isolated functionalities, such as code editing or online execution, without providing a comprehensive and integrated solution. Several systems lack essential features such as persistent storage, execution history, secure user management, and seamless language switching. Additionally, limited scalability and rigid system design make it difficult to extend these platforms with new features or technologies. These limitations highlight the need for a unified, mobile- friendly programming environment that combines cloud execution, multi-language support, and user-centric design.

The primary problem addressed in this work is the absence of a lightweight yet functional mobile programming platform that supports real-time code execution and personalized user experience. Most available tools either depend on high-resource desktop environments or provide incomplete functionality on mobile devices. Furthermore, the lack of persistent data management and structured execution feedback reduces their effectiveness for continuous learning and practice.

III. PROPOSED SYSTEM ARCHITECTURE

A. System Overview

The proposed mobile code editor is designed using a modular and scalable architecture to support real-time program writing and execution on Android devices. The system integrates a mobile frontend, application logic, cloud-based compilation services, and persistent data storage to deliver a lightweight yet functional programming environment. The architecture emphasizes portability, usability, and extensibility, allowing the system to support multiple programming languages and future feature enhancements without major redesign.

B. Architectural Layers

The overall system architecture is organized into the following key layers:

Presentation Layer:

The presentation layer provides an interactive and user-friendly mobile interface through which users can register, authenticate, write source code, select programming languages, and view execution results. This layer is responsible for handling user interactions and ensuring smooth navigation across different screens such as login and code editing.

Application Logic Layer:

The application logic layer manages the core functionality of the system. It handles user session management, code validation, language selection, and interaction control between the user interface and backend services. This layer acts as a bridge between the frontend and cloud execution services, ensuring that user requests are processed efficiently and securely.

Data Storage Layer:

The data storage layer manages persistent storage of user-related information such as authentication details, saved code files, and execution history. Structured data storage ensures consistency and reliability, enabling users to retrieve

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, March 2026
Impact Factor: 8.2

previous code submissions and maintain continuity across sessions. Secure data handling mechanisms are applied to protect user information and maintain data integrity.

Background Processing Layer:

The background processing layer handles asynchronous tasks such as saving execution history, managing API responses, and maintaining application performance. By processing non-blocking operations in the background, this layer ensures that the application remains responsive during code execution and data storage operations. It also supports scalability by allowing independent handling of background tasks as system usage grows.

The modular design of the proposed architecture enables independent development, maintenance, and scaling of individual components. This design approach ensures flexibility, improves system reliability, and allows seamless integration of additional features such as new programming languages, cloud synchronization, and advanced editor functionalities in future versions.

IV. METHODOLOGY

The development of the proposed mobile-based multi-language code editor was carried out using a structured and incremental approach to ensure systematic implementation and efficient integration of components. Initially, a detailed requirement analysis was performed to identify essential functional features such as user authentication, code editing, language selection, cloud-based execution, and result display. Non-functional aspects including performance, security, responsiveness, and scalability were also considered during planning. Based on these requirements, a modular layered architecture was designed to separate the user interface, application logic, cloud execution, and data management components, enabling better maintainability and organized system flow.

Following the design phase, the Android application was implemented with an intuitive interface to support seamless coding operations. Cloud-based compilation services were integrated through secure APIs, allowing user-submitted code to be transmitted to remote servers for execution and returning results in real time. Data storage mechanisms were incorporated to manage user credentials and execution history securely. Finally, extensive testing was conducted using multiple valid and invalid programs across supported languages to evaluate execution accuracy, response time, and application stability. The results confirmed that the system performs reliably under practical usage conditions.

V. RESULTS

The developed mobile code editor was evaluated under practical usage conditions using various programming scenarios. The system successfully executed valid programs and displayed meaningful error messages for syntactically incorrect or logically invalid code. Execution requests were transmitted to cloud-based compilation services, and results were returned in real time with minimal processing delay. The integration of asynchronous background handling ensured that the application remained responsive during execution processes. Performance observation confirms that the proposed system provides reliable execution support suitable for academic practice and learning environments.

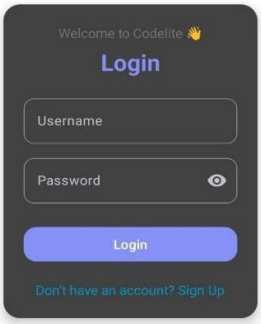

Fig. 1: Login Interface of CodeLite Mobile Application

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, March 2026
Impact Factor: 8.2

The Login Interface of the CodeLite mobile application serves as the primary entry point for users to access the system. It is designed with a clean and user-friendly layout that allows users to enter their registered email ID and password securely. The interface includes essential components such as input fields for credentials, a login button, and an option for new users to register. Proper input validation is implemented to ensure that incorrect or empty fields are handled appropriately, enhancing security and preventing unauthorized access.

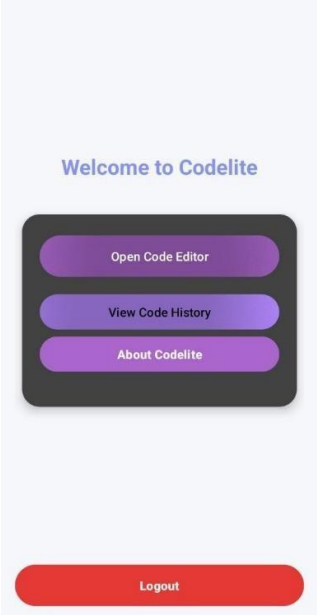

Fig.2: Menu Interface of CodeLite Mobile Application

The Menu Interface of the CodeLite mobile application provides users with structured navigation and access to the core features of the system. After successful login, users are directed to the menu screen, which presents organized options such as creating a new program, selecting a programming language, viewing saved programs, accessing execution history, and logging out. The interface is designed with a simple and intuitive layout to ensure smooth interaction and easy accessibility, even for beginner users.

The above figure illustrates successful compilation and execution of a sample program within the mobile interface. The output returned from the cloud server is displayed clearly to the user, demonstrating the effectiveness of remote execution integration.

VI. COMPARATIVE ANALYSIS

Feature	Traditional IDE	Existing Mobile Editors	Proposed System
Portability	Low	Moderate	High
Local Compiler Requirement	Required	Not Required	Not Required
Multi-Language Support	High	Limited	High
Cloud-Based Execution	No	Partial	Yes
System Resource Usage	High	Low	Low
User Accessibility	Desktop Only	Mobile	Mobile

The comparative analysis indicates that the proposed system provides enhanced portability, execution flexibility, and usability compared to conventional programming environments.

DOI: 10.48175/IJARSCT-31418

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, March 2026
Impact Factor: 8.2

VII. CONCLUSION

This paper presented CodeLite, a mobile-based multi-language code editor integrated with cloud-assisted execution services. The system eliminates dependency on local compilers and enables real-time program execution directly from Android devices. Experimental evaluation confirmed accurate output generation, minimal response delay, and stable performance. The modular architecture ensures scalability and maintainability, making the system suitable for modern programming education and portable development needs. The proposed solution demonstrates how cloud computing and mobile platforms can be effectively combined to create an accessible and efficient programming environment.

REFERENCES

[1] A. Sharma and R. Patel, "Cloud-based program compilation and execution for mobile applications," International Journal of Computer Applications, vol. 176, no. 32, pp. 12–18, 2020.

[2] P. Verma, A. Deshmukh, and R. Kulkarni, "Integration of online compiler APIs for mobile programming environments,"
International Journal of Advanced Computer Science and Applications, vol. 11, no. 6, pp. 245–252, 2020.

[3] M. Patel and K. Shah, "Design of a lightweight mobile code editor with execution support," Journal of Mobile Computing, vol. 9, no. 2, pp. 101–109, 2019.

[4] S. Ahmed and T. Williams, "Cloud-assisted development environments for learning-oriented programming platforms," IEEE Access, vol. 8, pp. 134210–134220, 2020.

[5] J. Brown and L. Wilson, "Mobile-friendly programming tools for beginner-level software education," Education and Information Technologies, vol. 26, no. 4, pp. 4567–4582, 2021.

[6] K. Mehta and A. Dubey, "A study on multi-language code execution using online compiler services," International Journal of Engineering Research and Technology, vol. 10, no. 5, pp. 88–94, 2021.