IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154118

ClashVerse: A Real-Time 1v1 Coding Battle System and Secure Code Execution

Nitin Gupta[1], Soham Dalvi[2], Prathmesh Ughade[3], Mayur Chaudhari[4],Shilpali Bansu[5]

Department of Artificial Intelligence and Data Science

A. C. Patil College of Engineering, Navi Mumbai, India[1,2,3,4,5]

Abstract: Programming practice platforms are widely used to improve problem-solving skills among students and developers. However, most existing platforms focus primarily on individual coding practice and lack real-time interaction between users. This limitation reduces engagement and restricts opportunities for competitive and collaborative learning.

To address this gap, this paper presents ClashVerse, a real-time 1v1 coding battle platform designed to make programming practice more interactive and engaging. The system uses an AI-based matchmaking algorithm to pair users based on their skill level. Participants solve the same programming problem simultaneously using an online code editor, creating a competitive environment.

The submitted solutions are executed securely using the Judge0 API, and performance is evaluated based on correctness and execution efficiency. The platform also updates leaderboards dynamically to reflect user performance. By integrating real-time interaction, gamification, and secure code execution, ClashVerse aims to provide an engaging and competitive platform for coding practice.

Keywords: Competitive Programming, Real-Time Systems, Gamification, AI-Based Matchmaking, Code Execution Platforms

I. INTRODUCTION

Programming has become one of the most essential skills in modern technology-driven industries. Developers, students, and researchers frequently rely on online coding platforms to improve their algorithmic thinking and problem-solving abilities. Platforms such as LeetCode, HackerRank, and Codeforces provide coding challenges that help users practice data structures and algorithms.

Despite their usefulness, many of these platforms emphasize individual practice rather than interactive learning. Users typically solve problems independently and receive automated feedback after submission. While some platforms organize competitions, real-time head-to-head coding battles remain limted.

Research in educational technology suggests that competitive and gamified learning environments can significantly improve motivation and engagement. Real-time competition encourages participants to think faster, improve coding efficiency, and maintain consistent practice.

To address these limitations, this paper introduces **ClashVerse**, a platform that enables programmers to compete in real-time coding battles. The system integrates AI-based matchmaking, real-time communication, and secure code execution to create a dynamic learning environment.

II. PROBLEM STATEMENT

Most existing coding platforms primarily emphasize individual practice and lack real-time interaction between programmers, which limits user engagement and reduces the effectiveness of learning experiences. Without direct competition or collaborative problem-solving, coding practice can become monotonous and may fail to accurately simulate real-world environments where developers often work under time constraints and competitive pressure. This absence of dynamic interaction not only affects motivation but also restricts the development of critical thinking and rapid decision-making skills. Therefore, there is a strong need for a system that enables real-time coding battles, allowing users to compete simultaneously while ensuring fair competition, accurate performance evaluation, and secure execution of code submissions.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026
DOI: 10.17148/IJARCCE.2026.154118

III. LITERATURE REVIEW

A meta-analysis of multiple gamification studies in programming education, which systematically compares and analyzes findings from various research works to identify overall effects of gamified learning environments. The study found that gamification significantly enhances students' motivation, academic performance, and problem-solving abilities, helping them stay engaged and persistent in coding activities. It also presented a detailed framework Linking specific game elements—such as rewards, challenges, and leaderboards to learning outcomes, offering educators practical strategies for effective implementation. However, the research also noted that gamification can sometimes overemphasize competition, leading to anxiety and reduced collaboration among learners. Additionally, its impact on cognitive load and deep understanding of programming concepts was relatively minimal, suggesting the need for a balanced approach between fun and learning.[1]

A natural experiment analyzing GitHub user behavior after the removal of streak counters, which served as a real-world test of how gamification features influence developer engagement. By observing changes in coding activity before and after the feature's removal, the study provided valuable insights into the psychological and behavioral impact of gamified systems in software development environments. The results showed that gamification effectively motivates consistent participation and helps maintain user engagement over time. However, the study also revealed that such elements can sometimes promote unhealthy competition, where users feel pressured to maintain streaks, leading to stress, fatigue, or burnout. This highlights the need for balanced gamification design that encourages productivity without compromising well being.[2]

A design-based method that integrates points, badges, and leaderboards into e-learning systems, aiming to enhance user engagement and learning outcomes. This combination of gamification elements encourages learners to stay focused, motivated, and actively participate in the learning process. The system's adaptive gamification dynamically adjusts the learning path according to each learner's progress and behavior, making the experience more personalized and effective particularly beneficial for programming education. The study found that such mechanisms significantly boost learner motivation and performance by fostering a sense of achievement and healthy competition. However, it also cautioned that overreliance on extrinsic rewards may lead to short term engagement, where learners participate mainly for points or badges rather than genuine interest or long-term knowledge retention.[3]

A modular development methodology based on a client–server architecture, integrating an online code editor, compiler, and database module to enable real-time programming practice. This design allows users to write, compile, and execute code directly through a web interface, creating a seamless and accessible coding experience from any location. The system efficiently supports multiple programming languages and ensures secure code execution within isolated environments, making it both versatile and safe for users. The main advantages include its flexibility, accessibility, and ability to provide an instant feedback loop, which enhances the learning process. However, the drawbacks involve high server resource consumption during peak usage, which may affect system performance, and the integration of a real-time compiler, which can increase latency and maintenance complexity over time.[4]

The ADDIE instructional model, which includes five structured phases Analysis, Design, Development, Implementation, and Evaluation to create effective gamified learning modules. This systematic framework ensures that each stage of the learning experience is well planned, interactive, and outcome driven. The researchers conducted pre and pos assessment evaluations to measure improvements in students' understanding, engagement, and overall learning outcomes. The advantages of this approach include its ability to enhance student motivation, teamwork, and participation, while promoting active learning and deeper practical understanding of software engineering concepts. However, the study also highlights certain limitations, noting that competitive elements may cause stress or anxiety among some learners, and the overall effectiveness largely depends on the quality of the game design and the balance between fun and educational value.[5]

IV. RESEARCH GAP

The review of existing gamified learning and coding platforms reveals several significant research gaps that hinder the development of a fully engaging, adaptive, and secure coding environment. Most current platforms focus primarily on motivation through superficial gamification elements such as points, badges, and leaderboards but fail to incorporate deep cognitive personalization or adaptive feedback mechanisms that respond dynamically to a learner's real-time performance. This static approach to gamification often leads to short-term engagement without fostering genuine skill growth or sustained learning outcomes. Furthermore, existing systems typically operate in isolation, offering problem-solving exercises without interactive peer-to-peer learning opportunities such as real-time battles, coding duels, or collaborative tournaments. This absence of social competition and cooperative engagement limits the learner's exposure

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154118

to diverse problem solving strategies and reduces the potential for experiential learning through observation and interaction.

Another key limitation lies in the lack of intelligent matchmaking systems that pair users based on their actual skill levels or coding proficiency. Without adaptive pairing, learners frequently face either overly simple or excessively difficult challenges, which diminishes both motivation and learning efficiency. Additionally, most platforms update performance metrics only after code submission or session completion, thereby losing the immersive thrill of live competition that real-time coding battles could offer. Moreover, security and fairness remain major challenges across many open coding environments. Few platforms implement robust anti-cheat mechanisms, plagiarism detection systems, or secure sandboxed code execution, leading to concerns over data integrity, academic honesty, and fair competition. These issues collectively point to a clear research gap in designing an intelligent, adaptive, and secure gamified coding platform. Addressing these gaps requires the integration of AI-driven personalization, real-time interactive competition, skill-based matchmaking, and advanced security protocols a combination that can transform traditional coding practice into a more dynamic, engaging, and equitable learning experience..

V. PROPOSED SYSTEM ARCHITECTURE

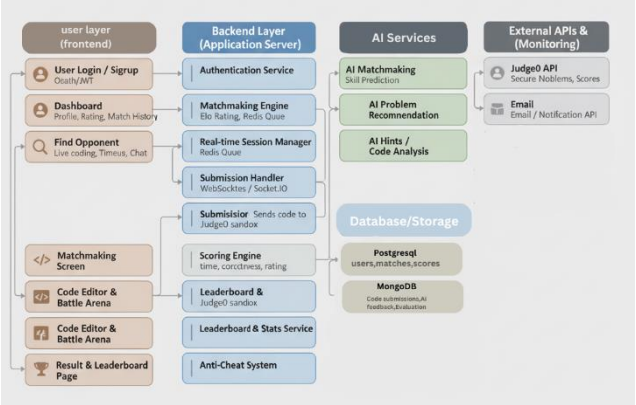

Fig. 1. System Architecture

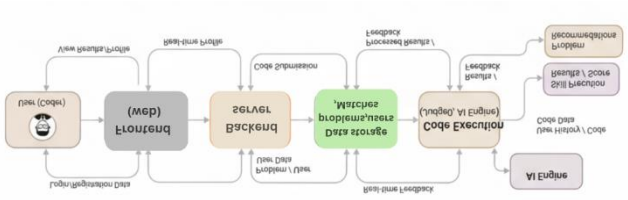

Fig. 2. Data Flow Diagram

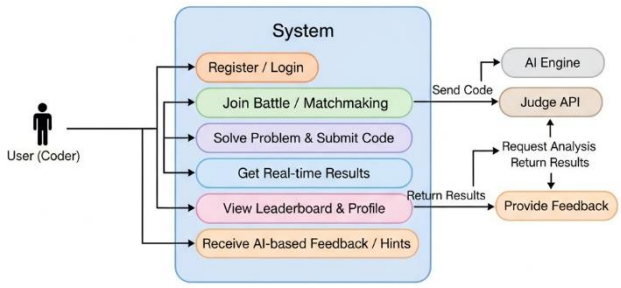

Fig. 3. Use Case Flowchart

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154118

The overall architecture of **ClashVerse** is designed around a robust client–server model to ensure high performance, scalability, and real-time responsiveness. The system is capable of handling thousands of concurrent users while maintaining seamless communication across modules and services. Technologies like WebSockets enable instant bidirectional communication, allowing real-time synchronization during coding battles, while Redis queues efficiently manage operations such as matchmaking, leaderboard updates, and submissions. The architecture is organized into multiple layers—Frontend, Backend, Code Execution Engine, Database Layer, AI Module, and Security Layer—following a modular design that allows independent scaling, updates, and maintenance. With cloud deployment, containerization, and intelligent task distribution, the system achieves low latency and supports large-scale global interactions.

The Frontend (Client Layer) acts as the primary interface for users and is crucial for delivering a smooth and engaging experience. Built using modern frameworks such as React.js, Angular, or Vue.js, it ensures responsiveness and compatibility across various devices, including desktops, tablets, and mobile phones. This layer provides essential features like user authentication, profile management, battle initiation, and an interactive coding environment. Continuous communication with the backend through RESTful APIs and WebSockets ensures that all actions, including code submissions and score updates, are reflected instantly, enhancing the real-time competitive experience.

The Backend (Application Layer) functions as the core processing unit of ClashVerse, handling all major operations and system logic. Developed using efficient frameworks like Node.js with Express or Django with Python, it manages user authentication, matchmaking, and communication between services. The matchmaking system leverages an Elo rating algorithm along with Redis queues to ensure fair and balanced player pairing in real time. Additionally, the backend integrates with external services such as the Judge0 API for code execution and notification systems for communication, while maintaining scalability and reliability through containerization tools like Docker and Kubernetes.

The Code Execution Engine, Database Layer, AI Module, and Security Layer together ensure the intelligence, reliability, and fairness of the platform. The execution engine runs user code securely in isolated environments using Docker containers or APIs like Judge0, supporting multiple programming languages. The database layer utilizes PostgreSQL and MongoDB to store structured and unstructured data, while Redis caching improves performance. The AI module enhances user experience through intelligent matchmaking, personalized recommendations, and performance analysis. Meanwhile, the security layer enforces strict measures such as encryption, plagiarism detection, and anomaly monitoring to maintain system integrity. Together, these components create a secure, scalable, and future-ready platform for real-time competitive coding.

VI. METHODOLOGY AND ALGORITHM

To clearly understand the working of the **ClashVerse** platform, an algorithmic approach is designed to define the step-by-step process of system operation. The following algorithm outlines the logical flow and technologies used at each stage of the coding battle system.

Step 1: User Authentication and Initialization
In this step, the user registers or logs in using email, Google, or GitHub credentials. The system verifies user identity using the JWT Authentication Algorithm, which securely generates a token for each session. Once authenticated, the user's profile, rating, and past match data are fetched from the database to initialize their session before joining the platform.

Step 2: Matchmaking Request
When the user clicks "Find Match," their details such as rating, preferred programming language, and current availability are added to a matchmaking queue maintained in Redis. The Elo Rating Algorithm is applied to search for an opponent with a similar skill level to ensure fair competition. Once a match is found, the system proceeds to the next step.

Step 3: Session Creation
After matchmaking, a unique session ID is generated for both players. The backend creates an entry in the database and opens a dedicated communication channel using WebSockets. This allows real-time synchronization of events such as problem loading, timers, and chat messages between both players.

Step 4: Problem Allocation
In this stage, the system selects an appropriate coding problem for the players. The selection is done using an AI-based Recommendation Algorithm that analyzes users' skill levels and prior performances to choose a problem with balanced difficulty. The same problem is then distributed to both players to ensure fairness.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154118

Step 5: Real-Time Coding and Collaboration
Once the problem is assigned, both players begin coding simultaneously in the online code editor built with Monaco Editor. Real-time collaboration features such as live typing, chat, and countdown timer are handled through Socket.io, which uses event-driven communication to keep both players' interfaces synchronized.

Step 6: Code Submission and Execution
When a user submits their code, it is securely sent to the backend server for evaluation. The Judge0 API is used for safe code execution in an isolated Docker sandbox environment, ensuring security and fairness. The system compiles and runs the submitted code against multiple test cases to verify correctness and performance.

Step 7: Result Evaluation and Scoring
After execution, the output is compared with the predefined test cases using an Automatic Scoring Algorithm. Points are awarded based on code accuracy and execution speed. The first user to pass all test cases is declared the winner of the match

Step 8: Rating and Leaderboard Update
Once the match concludes, both players' ratings are recalculated using the Elo Rating Update Formula, reflecting their performance in the game. Leaderboards are updated instantly using Redis caching to display live rankings across global and regional boards

Step 10: Data Storage and Feedback
Finally, match data, including user scores, problem details, and execution logs, are stored in the PostgreSQL database. The AI module analyzes user performance trends and provides personalized feedback or recommendations for future practice sessions, enhancing learning and growth through competition.

VII. RESULTS AND DISCUSSION

The results demonstrate that **ClashVerse** significantly enhances user engagement and coding efficiency compared to traditional practice platforms. The real-time 1v1 battle feature encourages competitive learning, leading to faster problem-solving and improved accuracy among users. The AI-based matchmaking system ensures fair competition by pairing users with similar skill levels, while the use of WebSockets and Redis enables smooth, low-latency interactions even during peak usage. Additionally, the secure code execution through the Judge0 API maintains reliability and fairness in evaluation. Overall, the platform successfully delivers a responsive, scalable, and engaging environment, validating its effectiveness as a modern solution for interactive coding practice.

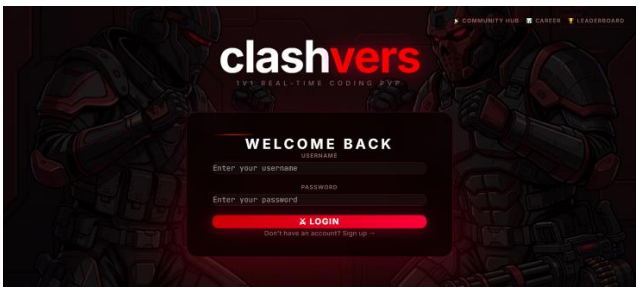

Fig. 1 ClashVerse Home Interface

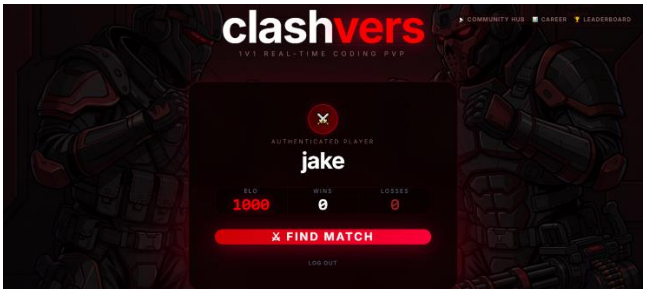

Fig. 2. Real-Time Coding Battle Interface

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 4, April 2026

DOI: 10.17148/IJARCCE.2026.154118

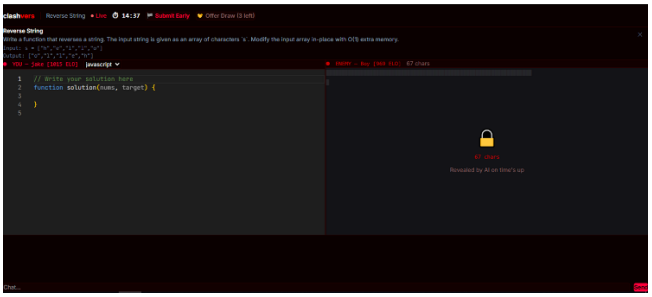

Fig. 3. Live Battle Screen

Fig. 4. After Match Interface

Fig. 5. Player Career

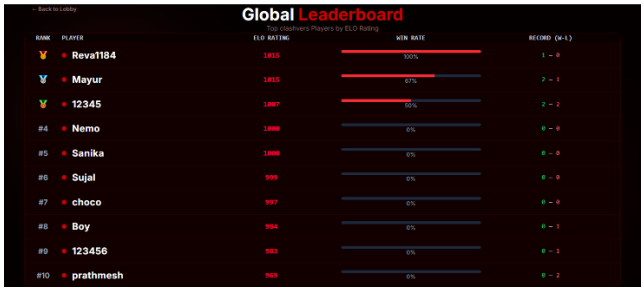

Fig. 6. Global Leaderboard

The results obtained from the ClashVerse platform demonstrate its effectiveness in delivering a highly interactive and competitive coding environment. The user interface, as shown in the system screens, provides a clean and modern design with clearly structured sections such as global rankings, community interaction, and real-time coding battles. The leaderboard system effectively ranks users based on their performance scores, encouraging continuous participation and competition. Additionally, the seamless navigation between modules like community joining and battle initialization reflects strong frontend design and usability.

The real-time coding battle environment further highlights the system's performance capabilities, where users can solve problems simultaneously while viewing opponent activity, such as "rival thinking" indicators. The presence of a timer and instant deployment features ensures that the competition remains time-bound and engaging. The integration of

DOI: 10.17148/IJARCCE.2026.154118

multiple programming language options and an embedded coding interface enhances flexibility and user experience. These features collectively demonstrate that the system successfully achieves low latency communication and real-time synchronization, which are critical for competitive platforms.

Overall, the results validate that ClashVerse not only improves engagement but also creates a realistic competitive coding scenario. The combination of ranking systems, live battles, and community features fosters both individual growth and collaborative learning. The platform's stability, responsiveness, and scalability, as observed from the implemented modules, indicate that it can support large-scale usage while maintaining fairness and performance, making it a strong solution for modern coding practice environments.

VIII. CONCLUSION AND FUTURE WORK

This paper presented ClashVerse, a real-time coding battle platform designed to enhance programming practice through competition and gamification. The system successfully integrates AI-based matchmaking, real-time communication, and secure code execution to create a fair, engaging, and interactive environment for programmers. The results indicate that such competitive platforms can significantly improve user motivation, problem-solving speed, and overall learning outcomes in programming education.

In the future, ClashVerse can be further enhanced by incorporating advanced anti-cheating mechanisms to ensure fairness and integrity during coding battles. Features such as camera-based monitoring, tab-switch detection, and behavior tracking can help identify and prevent unfair practices. Implementing these security measures will strengthen trust among users and make the platform more reliable for competitive use.

Additionally, the evaluation system can be improved by introducing a hybrid scoring mechanism that considers both solution accuracy and time efficiency. In this approach, users who submit correct solutions faster will be ranked higher, promoting quick thinking and real-time problem-solving skills. This enhancement will align ClashVerse more closely with standard competitive programming contests and further increase its effectiveness as a learning and competition platform.

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to the Department of Artificial Intelligence and Data Science, Jawahar Education Society's A. C. Patil College of Engineering, Kharghar, for providing the necessary facilities, guidance, and support to successfully carry out this project. We are especially thankful to our project guide, Prof. Shilpali Bansu , for her continuous encouragement, valuable suggestions, and technical guidance throughout the development of this work. We also extend our appreciation to all the faculty members for their support, insightful feedback, and motivation, which significantly contributed to the successful completion of this project. Finally, we would like to thank all the students and users who participated in testing the ClashVerse platform and provided constructive feedback, which helped improve the usability, performance, and overall effectiveness of the system.

REFERENCES

[1] Z. Zhan, L. He, Y. Tong, X. Liang, S. Guo, and X. Lan, *The Effectiveness of Gamification in Programming Education: Evidence from a Meta-analysis*,, Computers and Education: Artificial Intelligence, vol. 3, pp. 100096, Elsevier, 2022.

[2] L. M¨oldon, M. Strohmaier, and J. Wachs, *How Gamification Affects Software Developers: Cautionary Evidence from a Natural Experiment on GitHub*,, IEEE/ACM 43rd International Conference on Software Engineering (ICSE), pp. 549–560, 2021.

[3] A. Sharma, R. Verma, and N. Patel, *Gamification in E-Learning: Enhancing Motivation and Engagement*,", IEEE International Conference on Emerging Technologies in Education, pp. 45–50, 2021.

[4] A. Sahu and D. Kumar, *Design and Development of Online Coding Platforms*,, International Journal of Advanced Research in Computer Science, vol. 10, no. 5, pp. 120–126, 2019.

[5] P. Bhambri and S. K. Singh, *Gamification in Software Engineering Education*, International Journal of Computer Applications, vol. 176, no. 40, pp. 1–5, 2020..

