iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** IV **Month of publication:** April 2026

DOI: https://doi.org/10.22214/ijraset.2026.79263

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue IV Apr 2026- Available at www.ijraset.com

Coding Carnival: A Gamified Multi-Language Escape-Room-Inspired Platform for Immersive Programming Education

Madhu Gupta[1], Sameeksha Nayak[2], Mrudula Shenoy[3], Poonam Dharpawar[4]
Department of Data Science, Usha Mittal Institute of Technology, SNDT Women's University, Mumbai-400049, India

Abstract: ***Programming education continues to face challenges related to learner engagement, cognitive overload, and insufficient structured scaffolding, particularly in introductory courses. While gamification, escape-room-based learning, horror-themed immersive design, and adaptive educational systems have individually demonstrated potential to enhance motivation and mastery, existing platforms often address these components in isolation. The integration of immersive narrative engagement, structured pedagogical sequencing, and scalable backend architecture within a unified programming education framework remains underexplored. This paper introduces Coding Carnival, a modular, multi-language, web-based escape-room platform designed to teach programming concepts across Python, Java, and SQL domains. The system implements a structured instructional pipeline comprising concept explanation, worked example, and applied puzzle to ensure progression from comprehension to independent problem-solving. A relational database and RESTful API-based backend facilitate dynamic puzzle retrieval, server-side validation, and scalable content management. The platform features an immersive carnival-themed interface and a persistent in-game guidance agent that provides contextual hints to support learner progression. The development process followed a staged implementation and validation methodology, including frontend–backend integration, database schema design, functional testing, and system reliability verification. Results demonstrate successful dynamic content delivery, architectural modularity, and consistent performance across integrated puzzles. The proposed framework contributes a scalable escape-room-based programming education architecture that synthesizes immersive design, structured pedagogy, and modular system implementation. Future work will focus on empirical evaluation of learning outcomes and integration of adaptive learning mechanisms to enhance personalization and instructional effectiveness.***

Keywords: game-based learning, escape-room pedagogy, programming instruction, scalable web architecture, multi-language learning systems, structured pedagogical design, immersive educational environments, adaptive learning frameworks

I. INTRODUCTION

Programming remains the foundational component of modern computing curricula; however, introductory courses continue to face persistent challenges related to learner engagement, cognitive overload, and high attrition rates. Novice learners frequently struggle to translate conceptual explanations into applied problem-solving, particularly when instruction relies on lecture-centric delivery and static practice. Although numerous digital learning platforms exist, many fail to provide structured pedagogical progression or sustained motivational support. Consequently, scalable instructional frameworks that combine engagement-driven design with systematic concept reinforcement are needed.

Gamification and escape-room-based learning have been investigated as potential approaches to enhance learner motivation and participation in programming education [2], [9], [10]. Prior studies have demonstrated that escape room environments promote active problem-solving, collaboration, and increased engagement compared to traditional instruction [2], [7], [9]. Research on horror game design further indicates that immersive audio-visual elements and suspense-driven narratives can intensify user involvement and emotional investment [1], [3], [5]. In parallel, adaptive learning systems and tutoring frameworks have shown measurable improvements in programming mastery through personalized task recommendations and structured, tailored feedback [4], [8], [11]. Despite these advances, existing systems often address these components in isolation. Gamified platforms may lack modular backend scalability [2], adaptive systems are frequently detached from immersive environments [4], [8], and horror-themed educational games rarely incorporate structured pedagogical sequencing or multi-language extensibility [3], [5]. The integration of immersive narrative design, database-driven architecture, and structured programming instruction within a unified escape room framework remains insufficiently explored.

This study addresses this gap through the design and implementation of the Coding Carnival, a modular, multi-language, web-based escape room platform for programming education. The system integrates a structured instructional pipeline—concept explanation, worked example, and applied puzzle—to ensure progression from comprehension to independent reasoning through practice. The architecture employs a relational database and API-driven backend to enable dynamic puzzle retrieval, scalability, and cross-domain extensibility across Python, Java and SQL. Learners navigate the environment through a map-based interface within an immersive carnival theme. A persistent in-game guidance agent, Casper, provides contextual hints and progression support, functioning as a scaffolding mechanism, rather than a purely aesthetic element.

The main contributions of this study are as follows:

1) Design of a modular, database-driven escape-room architecture supporting multiple programming domains.
2) Integration of a structured pedagogical sequence within an immersive narrative framework to promote applied reasoning.
3) Implementation of dynamic backend-driven puzzle management with scalable content extensibility.
4) The incorporation of a guided narrative agent to provide contextual scaffolding within a gamified learning environment.

The subsequent sections of this paper present the related literature, describe the system architecture and implementation methodology, and outline the evaluation framework for assessing the effectiveness of the proposed approach.

II. METHODOLOGY

Each phase of development was treated as an experimental stage to validate functionality, usability, and system reliability. The materials, system components, and methodological procedures employed to design, implement, and integrate the gamified programming platform are as follows:

A. Materials Utilized in the Experiment

The development and integration of the Coding Carnival platform necessitated the following materials and technical resources:

1) Software Components

- HTML5, CSS3, and JavaScript for frontend development
- Node.js with the Express framework for backend development
- SQLite database for structured storage of puzzles and hints
- Visual Studio Code as the integrated development environment
- Postman for API testing
- Git and GitHub for version control and collaboration

2) Hardware Requirements

- Personal computer with a minimum of 8GB RAM
- Standard web browser (Google Chrome) for runtime testing
- Local development server environment

3) Dataset (Puzzle Repository)

A structured dataset comprising:

- 30 Java programming puzzles
- 20 SQL puzzles
- 30 Python puzzles

Each puzzle included:

- Tutorial explanation
- Worked example
- Independent challenge
- Expected output
- Difficulty level
- Hint levels

The dataset was maintained within a relational database schema to ensure scalability and controlled validation.

B. Step-by-Step Procedure Followed

The development process was executed in sequential experimental stages to guarantee controlled validation and progressive refinement of the system.

1) System Design and Learning Model Definition

A tutorial–example–puzzle framework was established to ensure pedagogical clarity. This structure aligns with escape-room-based programming education models that emphasize progressive challenge design and self-directed engagement [2], [9]. This structure ensured users first comprehended the concept before engaging in independent problem-solving. Logical separation between example and challenge tasks was validated.

Result*:*

An organized educational flow was achieved, minimizing redundancy between examples and puzzles.

2) Frontend Interface Development

The user interface was designed to emulate an escape-room-style progression system [2], [7]. Screens were developed for:

- Splash interface
- Authentication interface
- Storyline introduction
- Language selection
- Puzzle display

Interactive elements such as dynamic level loading, answer submission, hint display, and feedback messages were implemented using JavaScript.

Result*:*

A fully navigable prototype demonstrating gamified learning interactions was produced.

3) Backend Architecture Development

A RESTful backend server was implemented utilizing Node.js and Express frameworks. API endpoints were created for:

- Dynamic puzzle retrieval
- User response validation
- Quiz submission management

Puzzle data was stored in a structured SQLite database.

Result*:*

A functional backend capable of independently serving puzzle data and validating answers was established.

4) Database Schema Design and Population

The database schema was extended to incorporate:

- Tutorial text
- Example code
- Puzzle questions
- Starter code
- Expected output
- Difficulty level
- Hint levels

Data insertion scripts populated 100 structured puzzles across Java, Python, and SQL languages.

Result*:*

A scalable and modular puzzle repository was developed, eliminating hardcoded frontend data.

5) Frontend–Backend Integration

The frontend was adapted to retrieve puzzles through asynchronous API calls (fetch() method). Implemented changes included:

- Elimination of hardcoded puzzle objects
- Dynamic loading of puzzle data from backend endpoints
- Server-side answer validation
- Response-based user interface feedback

This separation of presentation and logic layers is consistent with modular learning architecture principles discussed in adaptive educational systems research [4], [8].

Result*:*

The system transitioned from a prototype to a database-driven architecture, ensuring separation of presentation and logic layers.

6) Functional Testing and Validation

Testing procedures included:

- API endpoint validation via Postman
- Browser-based runtime testing
- Edge case validation for incorrect answers
- Hint visibility assessment
- Level progression verification

Each puzzle was evaluated for:

- Correct output matching
- Appropriate hint escalation
- Accurate feedback messaging

Result*:*

All integrated puzzles demonstrated consistent behaviour across multiple test cycles.

C. Tools and Instruments Employed for Data Analysis

Although the project focused on system development rather than statistical experimentation, structured validation methods were applied to assess system reliability and functionality.

1) API Response Testing

Postman was utilized to validate JSON responses from backend endpoints. HTTP status codes (200, 404, 401) were monitored for accuracy.

2) Database Validation

SQLite browser and SQL queries verified:

- Correct data insertion
- Proper schema expansion
- Data retrieval accuracy

3) Runtime Debugging

Browser Developer Tools (Chrome DevTools) monitored:

- Network requests
- Console errors
- API latency

4) Performance Consistency Checks

- Multiple sequential puzzle loads were tested
- Concurrent API calls were observed
- Response time consistency was monitored

D. Reliability and Validity of Experiments

To ensure reliability:

- All puzzle validations were conducted server-side to prevent manipulation.
- Database entries were verified post-insertion.
- API endpoints were tested under repeated execution conditions.
- Frontend-backend integration was validated across multiple browsers.
- Logical separation between tutorials and challenges was maintained to avoid content duplication [2], [10].

The modular architecture permits the addition of puzzles without modification of core system logic, thereby preserving structural validity.

III. RESULTS AND DISCUSSIONS

The results are discussed in relation to system functionality, architectural scalability, pedagogical structure, and alignment with existing research in gamified programming education.

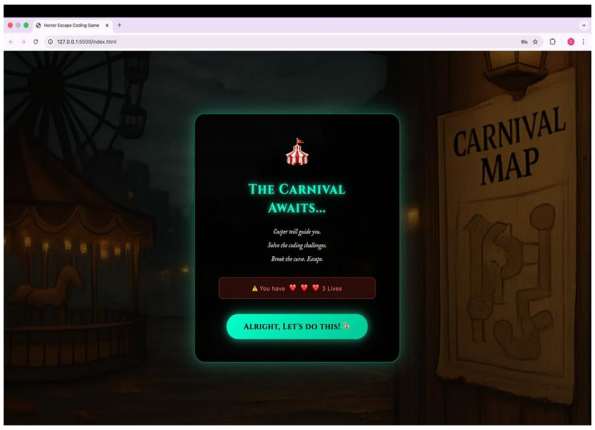


Figure 1. Splash Screen

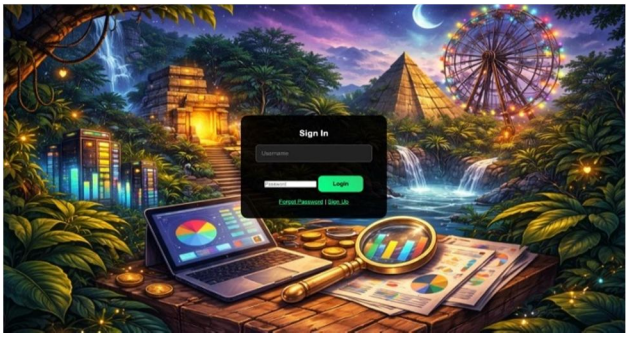


Figure 2. Login & Authentication

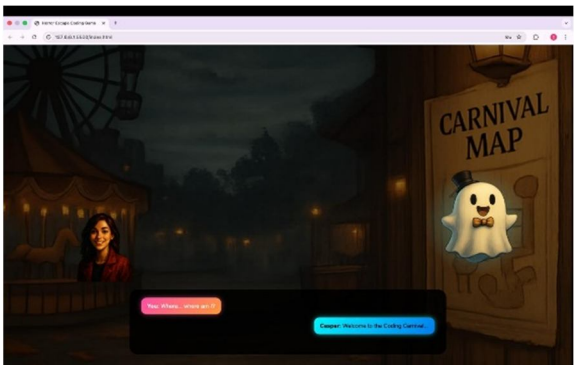


Figure 3. Guiding Agent – Casper

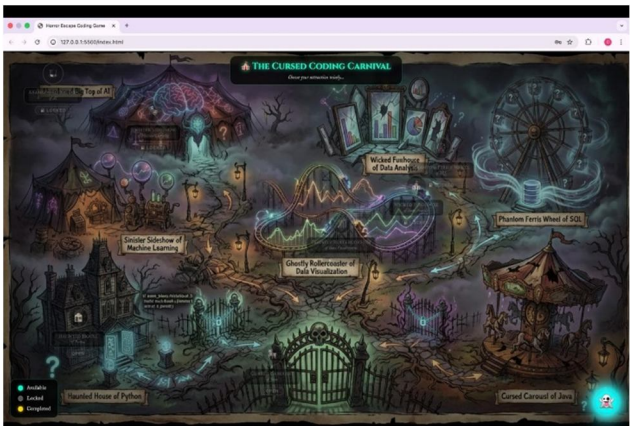


Figure 4. Carnival Map

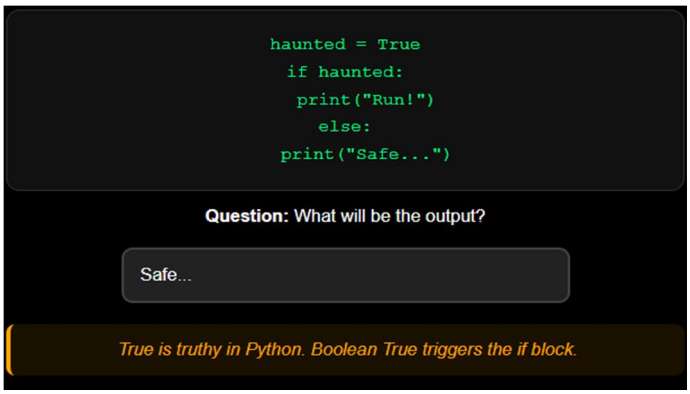


Figure 5. User Puzzle - Wrong Answer

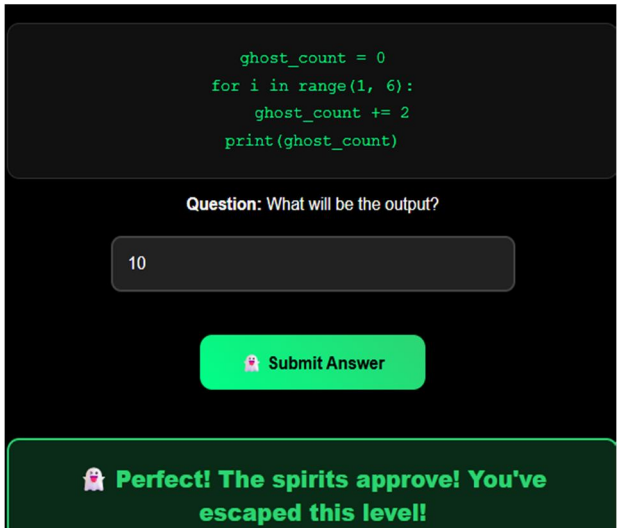


Figure 6. User Puzzle - Correct Answer

A. System Implementation Outcomes

The integration of frontend and backend components resulted in a fully operational, database-driven escape-room learning environment. The dynamic retrieval of puzzles via RESTful API endpoints effectively eliminated hardcoded frontend content, facilitating a structured separation between presentation and logic layers. All 100 structured puzzles across the Java, Python, and SQL domains were retrieved and validated through asynchronous requests without encountering runtime errors during repeated testing cycles.

Server-side validation ensured that user responses were processed independently of front-end manipulation, thereby enhancing reliability and structural integrity. API testing conducted using Postman confirmed consistent HTTP response behavior (200, 404, 401) and accurate JSON payload delivery. The monitoring of concurrent requests and sequential puzzle loading was undertaken to assess performance consistency, with no significant latency variation observed under local development conditions. These results demonstrate that the modular backend architecture supports scalable content management and controlled validation, addressing scalability concerns previously identified in escape-room-based programming systems [2], [9].

B. Pedagogical Structure Validation

The tutorial–example–puzzle pipeline was assessed for its logical separation and instructional coherence. Each instructional unit ensured that worked examples were distinct from independent challenge tasks, thereby minimizing repetitive pattern recognition. This structured sequencing directly addresses limitations observed in certain gamified programming platforms where examples and assessments are closely aligned, potentially encouraging memorization [10]. By enforcing conceptual explanation prior to independent problem-solving, the system operationalizes progressive challenge design, a core principle in escape-room-based educational models [2], [9]. Functional testing confirmed consistent hint escalation mechanisms across difficulty levels, allowing controlled scaffolding during puzzle attempts. This aligns with research indicating that guided support within immersive environments enhances engagement while reducing cognitive overload [7].

C. Immersive Framework and Narrative Integration

The carnival-themed interface, combined with the persistent in-game guidance, provides contextual progression cues and motivational scaffolding. Although quantitative emotional response measurement was not conducted in this phase, the structural integration of immersive narrative elements reflects findings from horror game design research, which emphasize the role of thematic immersion in sustaining engagement and emotional investment [1], [3], [5]. Unlike entertainment-focused horror systems, the immersive elements in Coding Carnival are systematically aligned with structured instructional flow. The guidance agent functions not merely as a decorative feature but as a scaffolding mechanism, offering contextual hints that support progression. This addresses the gap identified in prior research regarding the limited integration of immersive horror aesthetics with structured educational content [3], [5].

D. Architectural Scalability and Multi-Language Support

The relational database schema enabled storage and retrieval of structured puzzle components, including tutorials, examples, challenges, hints, and difficulty metadata. The modular design permits the addition of new programming languages or domains without modification of core system logic. This directly responds to limitations in several gamified programming platforms that are restricted to single-language environments [2], [10]. Furthermore, the separation of logic and presentation layers establishes a foundation for future integration of adaptive learning mechanisms. Prior studies on adaptive programming education emphasize the importance of modular architectures to support personalized task recommendation [4], [8], [11]. While adaptive algorithms were not implemented in this phase, the system architecture was intentionally designed to accommodate such extensions.

E. Discussion in Context of Existing Literature

The results demonstrate that Coding Carnival successfully integrates three research dimensions that are frequently treated independently in prior work:

- Escape-room-based programming education [2], [9]
- Immersive thematic engagement inspired by horror design principles [1], [3], [5]

Modular, scalable backend architecture compatible with adaptive learning systems [4], [8], [11] Whereas previous studies have validated engagement improvements in isolated contexts—either through gamification [6], [10] or horror-themed educational environments [5]—the present framework unifies immersive design, structured pedagogy, and database-driven scalability within a single system. Although large-scale empirical evaluation remains a future step, the functional validation conducted in this phase confirms that the platform operates reliably and maintains structural alignment with established educational design principles. The architectural results indicate readiness for subsequent experimental assessment involving learner performance metrics, engagement measurement, and adaptive model integration.

Overall, the implementation outcomes demonstrate that a modular, multi-language escape-room architecture can be successfully operationalized using a database-driven backend and structured pedagogical sequencing. The integration of immersive narrative scaffolding within a scalable system architecture addresses several gaps identified in existing literature and establishes a foundation for empirical validation in future research phases.

IV. CONCLUSION

This study addressed the need for a scalable and engaging framework for programming education by integrating escape-room-based gamification with structured pedagogical sequencing and modular backend architecture. Existing systems often treat immersive design, adaptive learning, and scalable content management as separate components. The objective of this work was to design and implement a unified, multi-language platform that combines structured instructional flow with immersive engagement mechanisms.

The proposed Coding Carnival platform incorporates a tutorial–example–puzzle pipeline to promote conceptual clarity and independent problem-solving across Python, Java, and SQL domains. A relational database and API-driven backend enable dynamic puzzle retrieval, server-side validation, and scalable content management. The inclusion of a guided narrative agent provides contextual scaffolding within an immersive environment, aligning engagement with instructional objectives.

The implementation results confirm the reliability, modularity, and extensibility of the system architecture.

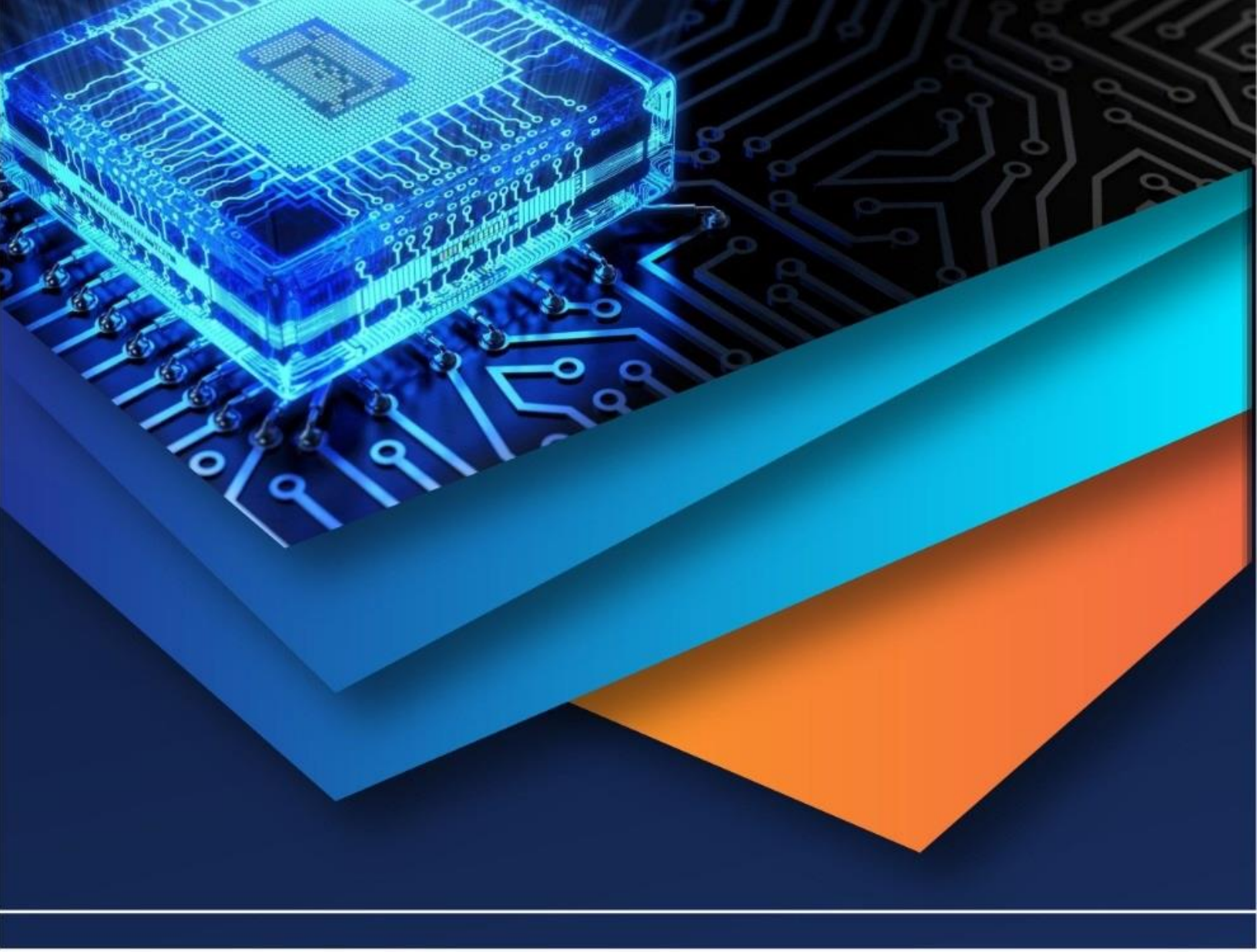
Future work will focus on integrating adaptive learning mechanisms to personalize puzzle difficulty based on user performance, incorporating AI-driven hint generation, and developing learning analytics modules to track learner progress. The platform can also be expanded to include additional programming domains and collaborative gameplay features, while cloud-based deployment would enable broader accessibility and scalability for educational use.

V. ACKNOWLEDGMENT

The authors thank Mrs. Poonam Dharpawar for her valuable academic guidance and constructive feedback throughout the development of this work. The authors also acknowledge the support provided by the Department of Data Science at Usha Mittal Institute of Technology, SNDT Women's University for access to institutional resources and technical infrastructure necessary for system implementation and testing.

REFERENCES

[1] A. Kumar, S. Sharma, and R. Gupta, "A research on improvements in horror game development," International Journal of Game Design and Technology, vol. 12, no. 3, pp. 145–158, 2023.
[2] N. Humble, P. Mozelius, and L. Sällvin, "You can't escape learning, but maybe you can get out of the room! Game-based learning for programming education," in Proc. Int. Conf. Interactive Collaborative Learning, 2021, pp. 234–241.
[3] Z. Zhang, "Analysis of the design aesthetics and player emotions of horror games," Journal of Game Studies and Design, vol. 8, no. 2, pp. 67–79, 2022.
[4] L. Na Nongkhai, J. Wang, and T. Mendori, "Development and evaluation of adaptive learning support system based on ontology of multiple programming languages," Education Sciences, vol. 15, no. 6, p. 724, 2025.
[5] L. Karvovskaya, J. Yeomans, and E. Rodenburg, "The data horror escape room game as a successful tool for RDM education and engagement," Liber Quarterly, vol. 35, no. 1, pp. 1–29, 2025.
[6] H. M. M. Ahmed, H. A. El-Sabagh, and D. Elbourhamy, "Effect of gamified, mobile, cloud-based learning management system (GMCLMS) on student engagement and achievement," Journal of Educational Technology Systems, vol. 54, no. 1, pp. 89–105, 2025.
[7] J. López-Belmonte, A. Segura-Robles, A. Fuentes-Cabrera, and M. E. Parra-González, "Evaluating activation and absence of negative effect: Gamification and escape rooms for learning," Computers & Education, vol. 146, p. 131-161, 2020.
[8] M. Kwak, J. Jenkins, and J. Kim, "Adaptive programming language learning system based on generative AI," Issues in Information Systems, vol. 24, no. 3, pp. 222–231, 2023.
[9] R. Queirós, C. Pinto, M. Cruz, and D. Mascarenhas, "A gamified educational escape rooms' framework for computer programming classes," IEEE Access, vol. 11, pp. 872–885, 2023.
[10] M. Pinto and T. Terroso, "Learning computer programming: A gamified approach," Education and Information Technologies, vol. 27, no. 4, pp. 123–140, 2022.
[11] M. Videnovik, T. Vold, L. Kiønig, A. Madevska Bogdanova, and V. Trajkovik, "Game-based learning in computer science education: A scoping literature review," International Journal of STEM Education, vol. 10, no. 54, pp. 23–40, 2023.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY