ISSN 2085-4579

CodeGuardians: A Gamified Learning for Enhancing Secure Coding Practices with AI-Driven Feedback

Raden Budiarto Hadiprakoso[1], Rudolf Paris Parlindungan Sihombing[1]
[1] Politeknik Siber dan Sandi Negara: Cryptography Engineering, Bogor, Indonesia
raden.budiarto@poltekssn.ac.id

Accepted on November 5th, 2024
Approved on December 10th, 2024

***Abstract*—This paper introduces CodeGuardians, a gamified platform designed to improve secure coding practices using AI-driven, real-time feedback. The platform focuses on key secure coding concepts, such as input validation, authentication, session management, and cryptography. Developed using the ADDIE (Analyze, Design, Develop, Implement, and Evaluate) instructional model, CodeGuardians enhances engagement and knowledge retention by incorporating interactive challenges. The AI component, powered by OpenAI, provides adaptive feedback on user-submitted code, helping users to learn secure coding practices more effectively. To assess its impact, a one-group pretest-posttest design was conducted. The results of a paired sample t-test showed a significant improvement in secure coding knowledge (t = 19.50, p = 0.048), confirming the platform's effectiveness. In addition, the system's usability was rated highly, with a score of 0.93 on the Computer System Usability Questionnaire (CSUQ), classifying it as "Very Good." The practical implications of this research suggest that CodeGuardians could be implemented in both educational and professional settings to enhance secure coding skills and reduce software vulnerabilities. From a theoretical standpoint, this study advances cybersecurity education by integrating AI-driven feedback into gamified learning environments. The research supports the theory that gamification improves engagement and learning retention, while also highlighting the value of adaptive technologies in addressing real-world security challenges. Future work will examine the long-term retention of knowledge and scalability across diverse learning environments.**

Index Terms*— *AI-Driven Learning; Cybersecurity Education; Gamification learning; Interactive Learning; Secure Coding

I. INTRODUCTION

The widespread dissemination of code snippets through platforms like StackOverflow has significantly contributed to the spread of insecure coding practices. This is particularly concerning given that empirical studies reveal 58.4% of answers on StackOverflow are outdated, with only a small fraction receiving necessary updates. Outdated answers on StackOverflow can propagate insecure coding practices by encouraging the use of deprecated libraries or unsafe methods, which can introduce vulnerabilities. Bei et al. [1] highlighted prevalent code license violations, identifying numerous instances of Creative Commons violations stemming from code copied from StackOverflow. Furthermore, Mareek et al. [2] found that 15.4% of security-related code in the applications they reviewed contained at least one insecure snippet, underscoring the urgency for more robust and secure coding practices. These issues emphasize the increasing need for solutions that can actively address and reduce the proliferation of insecure code in real-world applications.

As cybersecurity threats continue to grow in complexity and frequency, innovative educational approaches have emerged as a key strategy to combat these vulnerabilities. Gamification and serious games have proven to be effective tools for enhancing engagement and learning outcomes in cybersecurity contexts [3]–[6]. Gamification enhances learning through reward systems, interactive challenges, adaptive feedback, storytelling, and collaborative competitions, making security concepts more engaging. By incorporating game elements into secure coding education, developers can experience more interactive and engaging learning environments that directly address security risks while reinforcing secure coding principles. The integration of game mechanics into educational platforms has already shown success in raising awareness of cybersecurity threats, teaching best practices, and improving retention of critical security concepts.

Smith et al. developed "CyberAware," a gamified platform that educates users about common cyber threats, such as phishing and malware attacks [7]. Similarly, Jones and Wang introduced "SecureMe," a mobile game that enhances users' understanding of personal data protection and password security [8]. Both studies demonstrate how gamification can improve user engagement and retention in cybersecurity training. However, there is still a gap in addressing secure coding practices specifically through these methods, particularly given the

persistent challenges of insecure code discussed earlier.

For cybersecurity professionals, serious games offer immersive environments to practice and refine essential skills without real-world risks. Serious games help develop skills like intrusion detection, incident response, secure coding, and decision-making under pressure by providing realistic, risk-free simulation environments. Doe et al. introduced "CyberWarrior," a serious game that simulates network defense scenarios, enhancing decision-making skills related to intrusion detection and response [9]. In addition, Lee and Kumar's "PhishSim" helps users recognize and manage phishing attempts through adaptive learning [10]. These examples highlight the potential of gamified training solutions in professional cybersecurity settings. Williams et al. [11] and Patel and Singh [12] further emphasize the value of gamified competition in improving technical competencies through collaborative challenges like "HackathonQuest" and competitive environments such as Capture the Flag (CTF) competitions.

While gamification has been successfully applied in general cybersecurity education, its application to secure coding is relatively underexplored. In this paper, we explore "CodeGuardians," a serious game that leverages AI, specifically large language models (LLMs), to analyze code submissions and provide real-time feedback. This game covers critical secure coding practices, including input validation, authentication management, session management, and cryptography. Input validation prevents injection attacks, authentication management secures user access, session management protects active sessions, and cryptography safeguards data integrity and confidentiality. By incorporating AI-driven feedback mechanisms, "CodeGuardians" aims to create an engaging and interactive environment for learning and reinforcing secure coding practices, directly addressing the insecure coding issues identified earlier. This AI integration marks a significant advancement in the gamification of cybersecurity education, where personalized, adaptive learning can offer tailored feedback to users of varying skill levels.

Despite the growing interest in using AI generative models in gamification, their use in secure coding education remains limited. AI generative models, such as LLMs, offer the potential to dynamically create personalized challenges and feedback that adapt to the learner's individual progress and needs. This not only improves engagement but also ensures that the training is relevant to the learner's proficiency, addressing the limitations of static, non-adaptive learning systems. As this research explores the use of generative AI models within a gamified secure coding framework, pioneers new methods that bridge the gap between outdated learning approaches and interactive, AI-driven educational systems.

Recent research has explored the intersection of gamification and secure coding, highlighting various applications of game-based learning to enhance cybersecurity training. Altunel et al. [13] examined gamification in a large software company, showing that the incorporation of game elements improved developer engagement and adherence to secure coding standards, ultimately reducing security vulnerabilities in the codebase. Similarly, Selvi et al. [14] developed "EscapeScript," a gamified coding platform that used interactive, escape room-style challenges to teach coding skills, including security practices, to future coders. Both studies demonstrate the effectiveness of gamified learning for teaching secure coding, but they do not integrate AI for dynamic, personalized feedback.

Reinsch and Sanders [15] discussed the potential for gamification in expanding virtual learning environments for engineers and project managers, including secure coding practices. Their approach fostered collaborative learning and knowledge sharing through a gamified interface, which improved users' ability to retain critical security concepts. There are several studies that have shown the effectiveness of gamification in cybersecurity [16-19]. However, challenges such as the integration of emerging technologies like generative AI provide avenues for further research in gamified cybersecurity education.

Liu et al. [20] extended gamification with AI in their design of "SmartPal," an AI chatbot that provides real-time learning assistance, but their research focuses on general learning management systems (LMS) without addressing secure coding. Purwar et al. [21] introduced "CultureVo," a serious game utilizing generative AI to create personalized, interactive lessons, showing the potential of AI to dynamically adapt educational content based on learner progress, but this approach was used in cultural education, not cybersecurity or secure coding.

The novelty of the current research lies in the integration of AI generative models, specifically large language models (LLMs), into a gamified platform ”CodeGuardians" that provides real-time, AI-driven feedback tailored to secure coding practices. This approach directly addresses gaps in prior research by focusing on personalized, adaptive learning within the context of secure coding, something not fully explored in existing gamified or AI-powered educational platforms. This research differentiates itself by pioneering the use of AI in secure coding education, bridging the gap between static learning models and interactive, responsive AI-driven experiences

II. METHOD

This research uses the ADDIE learning model methodology. ADDIE is an acronym for Analyze, Design, Develop, Implement, and Evaluate. ADDIE is a methodology used to guide the formulation of training or education programs. Jamaluddin et al mentioned that ADDIE and SDLC can be combined because both have similarities that can make it possible to create software designs with educational

purposes [22]. The ADDIE model was chosen because of its structured, iterative framework that ensures the systematic development of effective educational programs, making it particularly suitable for gamified learning solutions like "CodeGuardians." Each phase—Analyze, Design, Develop, Implement, and Evaluate—provides a clear roadmap for addressing educational goals while allowing continuous improvement.

The research steps performed in this study can be seen as fig.1. The research was initiated at the analysis, design, and development steps in the form of designing the CodeGuardians application. At the analysis step, the functional and non-functional requirements of the CodeGuardians application were analyzed. The design step is the process of creating a prototype by paying attention to aspects of the application user's needs. The deployment step is the process of developing the prototype into an application. After these steps are completed, the application implementation step is performed as a secure coding web education media. At the evaluation step, testing is performed in the form of application testing and one group pretest-posttest design to answer the problem formulation and create conclusions.

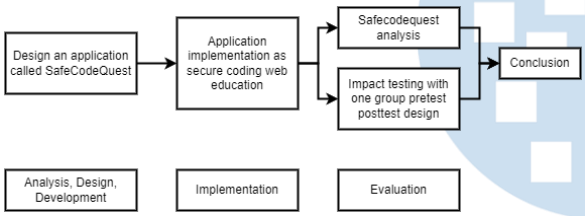

Fig. 1. Research Methodology

A. *Analysis*

The analysis phase focused on defining the functional and non-functional requirements of the CodeGuardians application. Functional requirements were gathered through a comprehensive review of existing secure coding educational tools, as well as interviews with cybersecurity professionals and educators. These inputs helped identify the core features required to meet user needs effectively. The functional requirements included:

1. User management: Sign up, sign in, logout, and profile management.
2. Educational tools: Challenge management, exam management, and a leaderboard system to encourage competition and engagement.
3. Scalability: Ability to handle multiple concurrent users without degradation in performance.

In terms of non-functional requirements, key aspects such as security, usability, and availability were prioritized. Usability testing focused on ensuring that the application is intuitive and accessible to users with varying levels of technical expertise. Additionally, security requirements included measures to protect user data and prevent unauthorized access, aligning with the core focus of teaching secure coding practices.

B. *Design*

During the design phase, a prototype of CodeGuardians was developed based on the requirements outlined in the analysis phase. The design process was iterative and included the creation of user personas and use case diagrams (as shown in Fig. 2), which illustrated how users would interact with the system.

The core game mechanics were designed to simulate real-world coding challenges. Users are presented with insecure code snippets and are required to correct vulnerabilities based on the lessons learned. The system includes three levels of difficulty, each corresponding to progressively more complex security issues. The design phase also involved sketching wireframes for the user interface (UI), emphasizing ease of navigation and clarity of instructions, especially for novice coders.

To ensure the robustness of the design, the platform architecture was outlined using UML diagrams. The frontend was designed using React.js, while the backend was built using the Flask Python framework. Data storage needs were addressed by utilizing a MySQL database, ensuring efficient management of user data and results.

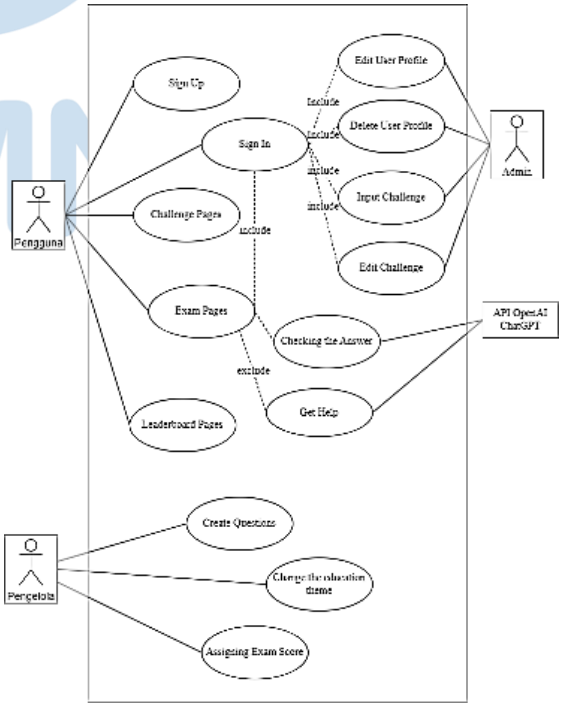

Fig. 2. Use Case Diagram

C. Development

The development phase involved coding the application based on the design specifications. The frontend, built using React.js, was integrated with Flask in the backend to ensure a seamless interaction between user actions and the system's response.

In addition to the technical development, this phase also included prompt engineering for the OpenAI API. Prompt engineering was critical for ensuring that the AI could accurately evaluate user-submitted code and provide real-time feedback. During development, prompts were iteratively tested to refine the accuracy and relevance of the feedback. For example, the OpenAI API was configured to provide specific feedback when users submitted incorrect or incomplete answers. Accuracy testing was conducted by inputting both correct and incorrect code snippets to ensure the AI could distinguish between secure and insecure coding practices. The prompts used are described in fig.3

Furthermore, automated testing frameworks were used to validate the system's performance. Unit tests were written for all major components to ensure the correctness of each functionality, while integration testing ensured smooth interactions between the frontend and backend.

> "You act as a code reviewer. Your job is to check the code inputted by the user based on the objective and key. When you check the code with the html type, ignore all case sensitive properties. If the objective and answer key are met and appropriate then the output is true, but if it is wrong then the output is false. If the output is false provide feedback that helps the user to improve their answer."

Fig. 3. Prompt engineering to OpenAI API

The prompt engineering process focused on designing the OpenAI API to function as a code reviewer, evaluating user-submitted code based on objectives and keys while providing actionable feedback for incorrect submissions. Testing involved iterative refinement using a diverse dataset of secure, insecure, and edge-case code snippets to ensure the AI accurately distinguished between correct and incorrect practices. Success criteria included achieving at least 90% accuracy in evaluations, providing clear and specific feedback aligned with secure coding principles, and robust handling of edge cases like incomplete or complex submissions. The system was also validated through integration testing to ensure seamless interactions between the frontend and backend, guaranteeing smooth user experiences in submitting answers and receiving feedback.

The game flows from the home page. When the user presses the start button, the user will be redirected to the challenge page. To take the exam, the user must first log in. After logging in, the user selects the level of the exam to be done. The user then reads the question description and question instructions, then corrects the wrong code in the game. If the answer is correct, the user will get points.

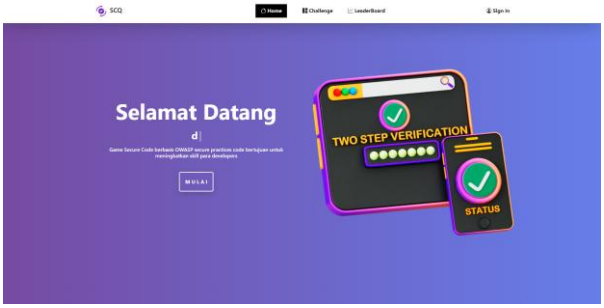

Fig. 4. Home Page

Fig. 4 shows the Home page of the game. When the app is first opened, there is a start button. If the start button is clicked, the player will be directed to the challenge page.

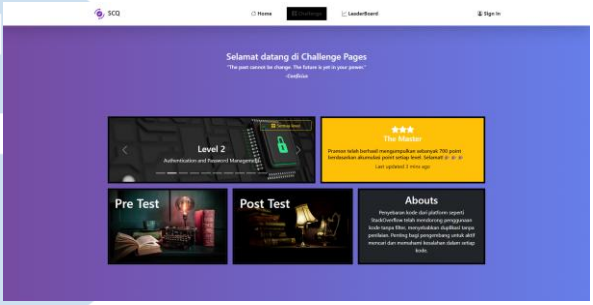

Fig. 5. Challenge Page

Fig. 5 displays the challenge page. On this page there is a menu for doing pretest, exam, and posttest. Players who want to take the exam, are required to log in first.

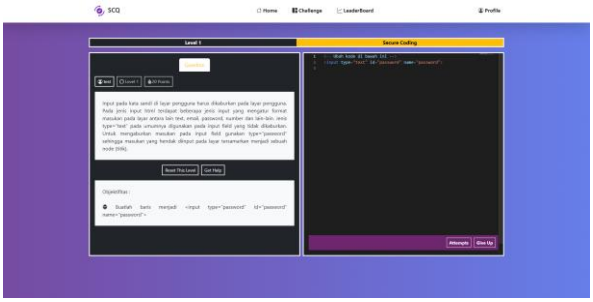

Fig. 6. Exam Page

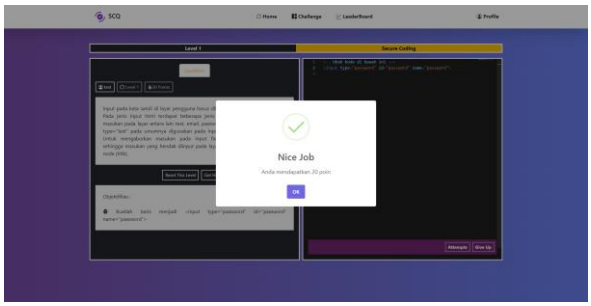

Fig. 7. Score Achievement

Fig. 6 is the level work page in the game. The level in Fig. 7 is the first level in the game. Users get instructions to disguise the password input on the web application frontend page by changing the type code from text to password. If the user succeeds in doing the instructions correctly, the user will get points as shown in Fig 7.

D. Implementation

The implementation phase involved deploying the CodeGuardians application for actual use by the research participants. Before full deployment, a pilot study was conducted with a small group of participants to identify any usability or performance issues. Feedback from the pilot users was used to make final adjustments to the application.

CodeGuardians was then made accessible to a larger group of participants, consisting of undergraduate students studying cybersecurity. The participants were tasked with completing various levels of coding challenges designed to test their secure coding knowledge. The application was hosted on a cloud-based server, ensuring high availability and scalability to accommodate multiple users The implementation process also included documentation and user guides to help users navigate the application and understand the learning objectives of each challenge.

E. Evaluation

The evaluation phase was crucial for assessing the effectiveness of the CodeGuardians application as a secure coding educational tool. This phase employed both formative and summative evaluation methods. Formative evaluation took place during the development and pilot stages, where feedback was gathered and incorporated into iterative improvements of the system.

For summative evaluation, a one-group pretest-posttest design was used [23]. Participants were tested on their secure coding knowledge before using CodeGuardians (pretest), and again after completing the challenges (posttest). Improvements in secure coding skills were measured by comparing pretest and posttest scores. Statistical analysis was performed to determine whether the improvements were significant. Additionally, user satisfaction surveys were distributed to gather qualitative feedback on the learning experience and usability of the system.

To assess the usability of the CodeGuardians application, the Computer System Usability Questionnaire (CSUQ), based on Cuasqui research [24], was administered to participants after they completed the challenges. The CSUQ measures system usability across three primary factors:

1. System Usefulness: How effectively the system helped participants accomplish the task of learning secure coding.
2. Information Quality: The clarity and relevance of the instructions, feedback, and information presented in the application.
3. Interface Quality: Participants' satisfaction with the interface design, including ease of navigation, visual layout, and overall aesthetics.

Participants rated each item on a 7-point Likert scale, with 1 being "strongly disagree" and 7 being "strongly agree." Key questions focused on the ease of use, whether the system helped them learn secure coding effectively, and whether the feedback provided by the AI component was clear and actionable.

III. RESULT AND DISCUSSION

A. One-group pretest-posttest

Hypothesis testing was conducted to measure the impact before and after the use of the application on users. The t-test is used to determine the decision on the impact of the application. The type of t-test used to analyze the impact of the application is paired sample t-test. Data on the hypothesis is taken through the pretest and posttest that has been collected. Respondents on the questionnaire were cadets of the National Cyber and Crypto Polytechnic who were taken by simple random sampling. The pretest and posttest questionnaires had 15 questions related to input validation, authentication management, session management, and cryptography practices. The questionnaire instrument was validated by experts in the field of cybersecurity and cryptography to ensure compliance with the target [10]. The following are the statistical results of the pretest and posttest calculations.

Table I shows the interpretation that the CodeGuardians application has a significant impact and a significant impact in improving users' ability to understand secure coding practices. With a significance level of $\alpha = 0.05$ and a t-value greater than the p-value, this result shows that there is an almost certain difference, so we reject H0 and accept H1.

TABLE I. T-TEST STATISTIC

Statistic	*Value*
t-value	19.50
Degrees of Freedom (df)	28
p-valued (2-tailed)	.048
Mean Difference	82.07
Standard Deviation	22.65
Standard Error Mean	4.28
Confidence Interval (95%)	[73.30, 90.84]

B. Usability Testings

To determine the score of the research results as the first step in measuring the usability of the CodeGuardians application, an assessment recapitulation was carried out. Each statement weight is recapitulated to get an assessment score based on equation (1).

TABLE II. USABILITY CALCULATIONS

Value	*Total*	*Score*
1	0	0
2	0	0
3	1	3
4	5	20
5	40	200
6	142	852
7	363	2541
Sum	551	3616

The highest usability score that can be obtained through equation 1 in this study is

$$highest\ score = 29 * 7 * 19 = 3857 \quad (1)$$

Furthermore, the usability score (x) is calculated based on the equation (2):

$$x = \frac{3616}{3857} = 0.93 \quad (2)$$

Based on the usability value (x) obtained from the CSUQ questionnaire, the CodeGuardians application received a value of 0.93. This value is still a quantitative value. Therefore, the usability value (x) is then interpreted into a predicate based on table 1 so that it can be concluded that the CodeGuardians application gets a predicate as Very Good category.

The implementation of the CodeGuardians application was evaluated through a comprehensive analysis that included both knowledge assessment and usability testing. The knowledge assessment involved hypothesis testing using a paired sample t-test to determine the effectiveness of the application in improving users' understanding of secure coding practices. The statistical results indicated a t-value of 19.50, with a degree of freedom (df) of 28 and a p-value of 0.048. The mean difference between the pretest and posttest scores was 82.07, with a standard deviation of 22.65 and a standard error mean of 4.28. The confidence interval (95%) ranged from 73.30 to 90.84, suggesting a significant improvement in users' knowledge after using the CodeGuardians application.

These results indicate that the CodeGuardians application had a statistically significant impact on enhancing users' understanding of secure coding practices. Given the p-value of 0.048, which is below the significance level of $\alpha = 0.05$, the null hypothesis (H0) is rejected, and the alternative hypothesis (H1) is accepted. This means that there is a significant difference between the pretest and posttest scores, demonstrating that the application effectively improved users' knowledge in areas such as input validation, authentication management, session management, and cryptography practices.

This finding aligns with other research in the field, which has shown that serious games and gamified applications can be effective tools for improving cybersecurity education. For example, Selvi et al. found that gamified learning environments significantly enhanced participants' ability to identify and respond to cyber threats [25]. Similarly, J. Lee, and A. Kumar. reported that serious games like "CyberWarrior" led to substantial improvements in cybersecurity skills among participants [26].

In addition to knowledge assessment, the usability of the CodeGuardians application was evaluated using the CSUQ. The usability score was calculated based on users' responses, resulting in a final usability value (x) of 0.93. This value was then interpreted using a predefined predicate table, which classified the application in the "Very Good" category.

The high usability score indicates that the CodeGuardians application is user-friendly, efficient, and provides a satisfying user experience. This is crucial for educational tools, as a positive user experience can significantly influence engagement and the effectiveness of learning. According to Nielsen, usability is a key factor in the success of interactive systems, particularly in educational contexts [26]. The usability score obtained for CodeGuardians suggests that the application not only facilitates learning but does so in a manner that is accessible and enjoyable for users.

Moreover, the strong usability performance of CodeGuardians is consistent with findings from similar studies. For instance, Zandvakili emphasized the importance of usability in gamified learning platforms, noting that applications with high usability scores tend to achieve better educational outcomes [27]. The "HackathonQuest". also highlighted the role of usability in ensuring the effectiveness of collaborative cybersecurity training tools [11].

The results from both the knowledge assessment and usability testing suggest that the CodeGuardians application is an effective tool for improving cybersecurity education. The significant improvement in knowledge, coupled with the high usability score, underscores the potential of gamified applications in this field.

The results of the study, while promising, have several limitations that must be considered when interpreting their applicability. First, the respondents were cadets from the National Cyber and Crypto Polytechnic, a specialized group with likely higher baseline knowledge of cybersecurity compared to the general population. This limits the generalizability of the findings to other user groups, such as novice developers or individuals without formal cybersecurity training.Second, the pretest and posttest questionnaires, while validated by experts, focused on specific secure coding practices (input validation, authentication management, session management, and cryptography). This narrow focus may not reflect a comprehensive understanding of all secure coding principles or their application in broader contexts.

IV. CONCLUSION

The research conducted on the CodeGuardians application has yielded several important conclusions regarding its effectiveness as a cybersecurity education tool. Firstly, the knowledge assessment results demonstrated that the application significantly improves users' understanding of secure coding practices. The paired sample t-test revealed a statistically significant increase in knowledge from pretest to posttest scores, indicating that CodeGuardians effectively enhances users' skills in critical areas such as input validation, authentication management, session management, and cryptography practices. This significant improvement validates the application's potential as an effective educational tool in cybersecurity training. Secondly, the usability evaluation of CodeGuardians, assessed using the CSUQ, resulted in a high usability score of 0.93, categorizing the application as "Very Good." This suggests that CodeGuardians is not only effective in delivering educational content but also provides a user-friendly, efficient, and satisfying experience. The high usability is crucial for ensuring that users remain engaged and motivated throughout the learning process, thereby maximizing the educational benefits of the application.

Overall, the findings indicate that CodeGuardians is a well-designed tool that effectively supports the learning of secure coding practices through an engaging, gamified approach. The significant impact on knowledge acquisition, combined with its excellent usability, positions CodeGuardians as a valuable resource for cybersecurity education. Future research should explore the long-term retention of knowledge gained through CodeGuardians and its applicability in diverse educational contexts.

REFERENCES

[1] W. Bai, O. Akgul and M. L. & Mazurek, "A qualitative investigation of insecure code propagation from online forums," IEEE Cybersecurity, 2019.

[2] M. Maarek, L. McGregor, S. Louchart and R. & McMenemy, How could serious games support secure programming? Designing a study replication and intervention, In 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW) (pp. 139-148): IEEE, 2019, June.

[3] S. Alma'ariz, R. B. Hadiprakoso, Girinoto and N. Qomariasih, "Soceng Warriors: Game-Based Learning to Increase Security Awareness Against Social Engineering Attacks," 2022 IEEE 8th Information Technology International Seminar (ITIS), Surabaya, Indonesia, 2022, pp. 124-129, doi: 10.1109/ITIS57155.2022.10009041.

[4] N. Katsantonis, I. M. Mavridis and D. & Gritzalis, "Evaluation of HackLearn COFELET game user experience for cybersecurity education," International Journal of Serious Games, vol. VIII, no. 3, pp. 3-24, 2021.

[5] Kublik, Sandra and a. S. Saboo, "GPT-3:," in The Ultimate Guide to Building NLP Products with OpenAI API, Packt Publishing Ltd, 13 Feb., 2023.

[6] S. Tingiris and B. Kinsella, in Exploring GPT-3: an unofficial first look at the general-purpose language processing API from OpenAI, Packt Publishing Ltd, 2021.

[7] J. Smith, A. Johnson, and L. Davis, "CyberAware: A Gamified Approach to Cybersecurity Education," IEEE Transactions on Learning Technologies, vol. 15, no. 2, pp. 123-134, 2022.

[8] M. Jones and Y. Wang, "SecureMe: Enhancing Personal Data Protection through Mobile Gaming," in Proceedings of the 2021 International Conference on Cybersecurity, pp. 45-52, 2021.

[9] A. Doe, K. Lee, and S. Patel, "CyberWarrior: Developing Cyber Defense Skills through Serious Gaming," Journal of Cybersecurity Education, vol. 10, no. 1, pp. 89-102, 2020.

[10] H. Lee and R. Kumar, "PhishSim: An Adaptive Serious Game for Phishing Awareness Training," IEEE Security & Privacy, vol. 19, no. 3, pp. 58-66, 2021.

[11] T. Williams, E. Clarke, and D. Nguyen, "HackathonQuest: Collaborative Learning in Cybersecurity Education," in Proceedings of the 2022 ACM Conference on Innovation and Technology in Computer Science Education, pp. 210-216, 2022.

[12] S. Patel and A. Singh, "The Role of Capture the Flag Competitions in Cybersecurity Skill Development," International Journal of Information Security, vol. 14, no. 4, pp. 321-330, 2021.

[13] H. Altunel, B. Say, M. Kosa, and M. Koca-Atabey, "Evaluation of an industrial case of gamification in software quality improvement," repository.bilkent.edu.tr, 2023. [Online]. Available: https://repository.bilkent.edu.tr/items/5909182a-a197-4b5e-a2fd-663c3a1e577b.

[14] K. Selvi, M. Shrinidhii, and G. Mounika, "EscapeScript: Gamified coding platform for empowering future coders," 2024 2nd International Conference on Advances in Computing, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10625310/.

[15] R Rachel and S. Emma, "Engineering and Project Manager Collaboration Expands Virtual Learning ,"2022. [Online]. Available: https://commons.erau.edu/pr-discovery-day/2023/presentations/40/.

[16] L. Garcia, M. Torres, and P. Martinez, "Evaluating Educational Outcomes of Cybersecurity Serious Games: A Systematic Review," Computers & Security, vol. 102, pp. 1-15, 2021.

[17] R. Thompson and J. Evans, "Assessing the Effectiveness of Gamified Cybersecurity Training Programs," IEEE Access, vol. 9, pp. 45678-45690, 2021.

[18] K. Miller and D. Brown, "Challenges in Developing Scalable Cybersecurity Gamification Solutions," IEEE Computer, vol. 54, no. 7, pp. 29-37, 2021.

[19] D. Liu, R. Zandvakili, A.T. Li, and R. Santhanam, "Design and evaluation of a gamified generative AI chatbot for Canvas LMS courses," International Conference on Human-Computer Interaction, 2024. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-031-61953-3_29.

[20] A. Agarwala, A. Purwar, and V. Rao, "CultureVo: The serious game of utilizing gen AI for enhancing cultural intelligence," arXiv preprint, 2024. [Online]. Available: https://arxiv.org/abs/2407.20685.
[21] S. Davis, H. Clark, and L. Rodriguez, "Future Directions in Cybersecurity Education: Integrating VR and AI in Serious Games," Journal of Educational Technology & Society, vol. 24, no. 3, pp. 150-162, 2021.
[22] R. A. Jamaluddin, H. Hasan, W. Fatimah, W. Ahmad and a. J. S., "Developing Interactive E-learning Content: A Subject Matter Expert Perspective," International Journal of Infrastructure Research and Management, vol. 11, no. 1, pp. 36-52, 2023
[23] .W. William and H. & Hita, "Mengukur Tingkat Pemahaman Pelatihan PowerPoint Menggunakan Quasi-Experiment One-Group Pretest-Posttest," Jurnal SIFO Mikroskil, vol. I, no. 20, pp. 71-80, 2019.
[24] P. J. Cuasqui, "Development and evaluation of usability of a gamified web application for basic computer education," Latin-American Journal of Computing, vol. 7, no. 1, pp. 138-151, 2020.
[25] Selvi, M. Shrinidhii, and G. Mounika, "EscapeScript: Gamified Coding Platform for Empowering Future Coders," 2024 2nd International Conference on Advances in Computing, Bengaluru, India, 2024, pp. 123-129. doi: 10.1109/ICAC42432.2024.10625310.
[26] J. Lee, and A. Kumar, "PhishSim: A Serious Game for Training Users to Detect Phishing Attacks," 2023 IEEE International Conference on Cybersecurity and Privacy, Singapore, 2023, pp. 567-573. doi: 10.1109/ICCP56682.2023.10082345.
[27] R. Zandvakili, D. Liu, A. T. Li, and R. Santhanam, "Design and Evaluation of a Gamified Generative AI Chatbot for Canvas LMS Courses," in Proceedings of the International Conference on Human-Computer Interaction, London, U.K., 2024, pp. 382-395. doi: 10.1007/978-3-031-61953-3_29.