Journal of Technology and Informatics (JoTI)
Vol. 8, No. 1, Bulan April Tahun 2026
P-ISSN 2721-4842
E-ISSN 2686-6102

Web-Based Automatic Code Evaluation System Using Claude AI for Programming Education

Paulus Lucky Tirma Irawan[1*], Windra Swastika[2]
[1*,2] Department of Informatics Engineering, Ma Chung University, Malang 65151, Indonesia
email: paulus.lucky@machung.ac.id[1*], windra.swastika@machung.ac.id[2]

Article Information

Article History:
Received : 5 December 2025
Revised : 6 March 2026
Accepted : 26 March 2026
Published : 27 April 2026

***Correspondence:**
paulus.lucky@machung.ac.id

Keywords:
Artificial Intelligence, Automatic Evaluation, Claude AI, Feedback System, Programming Learning

10.37802/joti.v8i1.1305

Journal of Technology and Informatics (JoTI)
P-ISSN 2721-4842
E-ISSN 2686-6102
https://e-journals.dinamika.ac.id/index.php/joti

***Abstract*:**

Algorithm and Programming learning face challenges in providing fast and personalized feedback to students. The manual evaluation process conducted by lecturers requires considerable time, hindering students' iterative learning process. This study aims to develop a web submission platform prototype with automatic feedback based on Claude AI to support and enhance the programming learning process. The research method employs a Research and Development (R&D) approach with four stages: needs analysis, system design and planning, platform implementation, and testing and evaluation. The platform was developed using a PHP backend, MySQL database, and Claude AI integration through RESTful web services with a cascading AI evaluation strategy. Evaluation was conducted on 9 students with 39 submissions for three Java assignments with different difficulty levels. Results show the system successfully provides high-quality feedback with an average response time of 2.8 seconds and 100% evaluation success rate. Score distribution shows average improvement from the first assignment (82.3) to the third assignment (87.1), indicating a positive trend in iterative learning. A satisfaction survey of 8 respondents shows the system interface is user-friendly, and AI feedback helps identify syntax and program logic errors. Students made an average of 3.2 attempts per assignment, demonstrating high engagement in the learning process.

INTRODUCTION

Algorithm and Programming courses constitute an essential foundation in the Informatics Engineering curriculum, aimed at developing students' logical and computational thinking abilities. However, traditional learning implementation faces significant constraints that hinder optimal learning objective achievement. Limited fast and personalized feedback becomes a primary challenge in the learning process, where students often experience difficulties understanding errors in their program code deeply and in a timely manner [1], [2], [3]. The manual evaluation process conducted by lecturers requires considerable time to

provide feedback on the source code created by students. This condition results in delayed constructive feedback delivery, hindering the iterative learning process that should occur dynamically and responsively. Previous research demonstrates that automated assessment systems can provide faster and more consistent feedback compared to traditional manual assessment [4], [5]. Students who do not immediately understand errors in their programming assignments tend to repeat the same mistakes, ultimately negatively impacting learning outcomes and overall algorithm concept comprehension [[6][7].

The digital transformation era in education demands innovations that leverage advanced technology to improve the efficiency and effectiveness of teaching and learning. Study of automated summative assessment for Python and C++ code, highlight that automated assessment tools have significant potential to support programming education through more structured, objective feedback [8]. Artificial intelligence, especially Claude AI from Anthropic, provides solutions to address programming learning challenges through code analysis and constructive, natural-language feedback. The need for responsive and adaptive learning systems becomes increasingly urgent, considering student ability heterogeneity and diverse learning styles [9], which affirm that automatic feedback systems can provide consistent, objective, and continuously available evaluation, enabling students to learn at their own pace while receiving quality guidance. This approach not only accelerates the evaluation process but also enables students to understand their errors deeply through detailed explanations and specific improvement suggestions [10][11].

A web platform with Claude AI-based automatic feedback blends advanced technology and best teaching practices in programming [12][13][14]. Souza, Felizardo, and Barbosa [15] found in their review that good programming assessment tools give feedback on both code output and quality factors such as readability, maintainability, and coding standards. This research aims to build a platform that automatically evaluates and supports student programming learning. The main goals are: develop a web platform with Claude AI to give automatic feedback on programming tasks, test its impact on student outcomes, and measure student satisfaction with the feedback system.

METHOD

This research employs a research and development (R&D) approach with a mixed-method approach combining quantitative and qualitative methods to develop and evaluate a web submission platform with Claude AI-based automatic feedback. The R&D approach was chosen as it aligns with research objectives that not only develop educational technology products but also test their effectiveness in real learning contexts [16][17]. The research was conducted in two main phases: the system development phase and the small-scale implementation evaluation phase. The development phase uses an iterative design approach, enabling continuous improvement based on user testing feedback. A small-scale implementation evaluation phase involves 9 students to test the platform's basic functionality.

Research Stages

This research was conducted through four interconnected and continuous main stages. Each stage has specific objectives, implementation methods, and clear success indicators to ensure research result quality and validity.

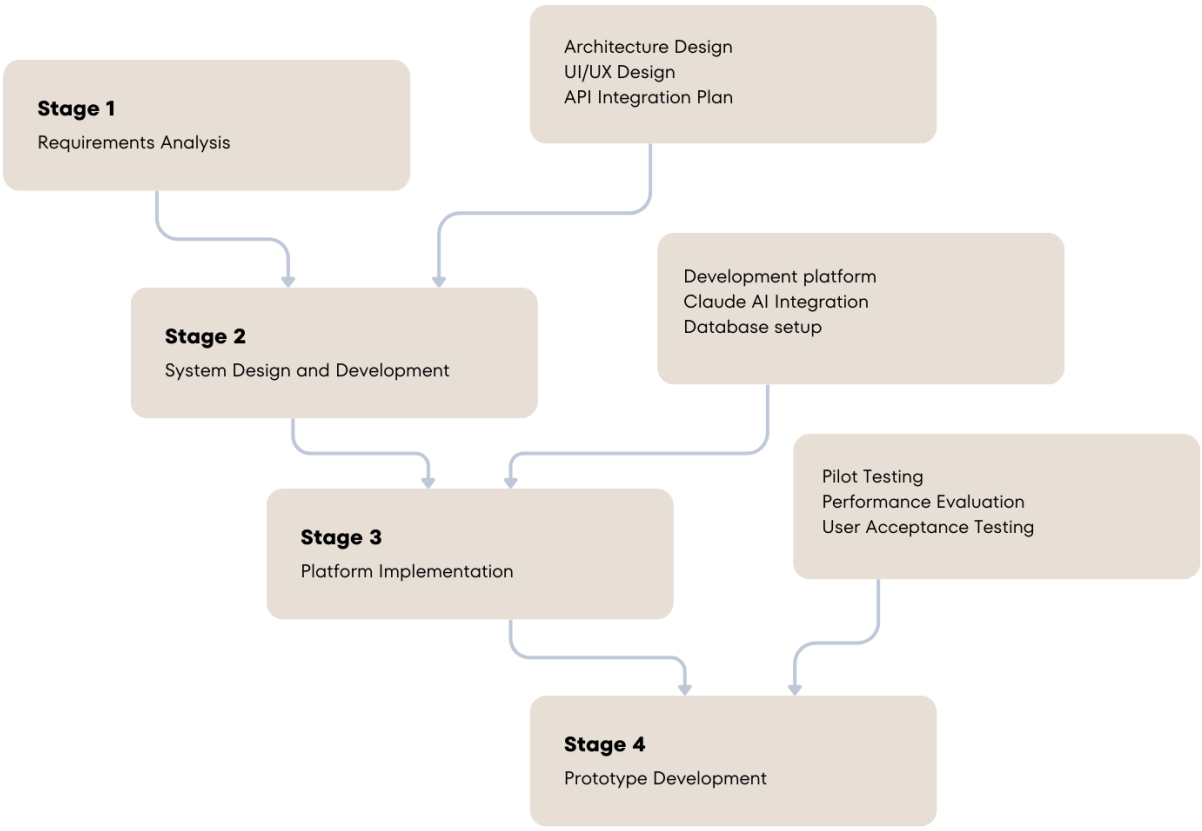

Figure 1. Research Stages

The research stages shows in Figure 1 begin with needs analysis through direct observation by Algorithm and Programming course lecturers. This stage is used to understand teaching and learning process characteristics and constraints faced in feedback provision. The next stage is System Design and Development. System architecture design is created using a user-centered design approach with the MVC (Model-View-Controller) model [18][19]. Design includes database, user interface, and Claude AI API integration. To systematically address the constraints identified during the preliminary observation, the system's specifications are delineated into comprehensive user and functional requirements, as presented in the subsequent tables. As delineated in Table 1 and Table 2, these structured requirements establish a clear foundation for the system's architecture, ensuring that the platform not only accommodates the specific interactive needs of both students and lecturers but also possesses the essential technical capabilities to execute automated, AI-driven code evaluations effectively.

The third stage is Platform Implementation. Development uses agile methodology utilizing coding AI tools. The KodePy platform employs a user-centered Model-View-Controller (MVC) architecture, establishing a clean separation between the responsive frontend (for students, lecturers, and administrators) [20] and the PHP/MySQL backend. A pivotal component is the RESTful AI integration module, which implements a cascading evaluation strategy to ensure robust performance. Submissions are primarily evaluated by Claude 3 Haiku, with OpenAI GPT-4o-mini as a fallback and a local Basic Evaluator as the final safety net. This multilayered approach ensures continuous availability, optimal cost efficiency, and a 100% evaluation success rate.

Table 1. User Requirement Analysis

Actor	Requirement ID	Requirement Description
Student	UR-01	students must be able to access active assignment pages and submit their programming code.
	UR-02	Students must be able to receive and view automated evaluation scores, feedback, and suggestions generated by the AI.
	UR-03	Students must be able to make multiple submission attempts for an assignment to facilitate an iterative learning process.
Lecturer	UR-04	Lecturers must be able to create, edit, and delete programming assignments within the system.
	UR-05	Lecturers must be able to view all student submissions and monitor the AI-generated feedback and scores.
	UR-06	Lecturers must have the capability to provide manual, personalized feedback and override the AI-generated scores with their own evaluations.

Table 2. Functional Requirement Analysis

Requirement ID	Requirement Description
FR-01	The system shall be developed as a responsive web platform utilizing a PHP backend and a MySQL database.
FR-02	The system must integrate with Claude AI via RESTful web services to provide automated code analysis and feedback generation.
FR-03	The system must implement a cascading AI evaluation strategy, utilizing Claude 3 Haiku as the primary service, OpenAI GPT-4o-mini as a fallback, and a Basic Evaluator as a final safety net.
FR-04	The automated evaluation must parse code based on a multi-criteria framework: Syntax Correctness, Logic Correctness, Code Structure, Best Practices, and Efficiency.
FR-05	The system must generate natural language feedback that identifies errors and provides actionable improvement suggestions.
FR-06	The system shall provide dedicated dashboards for monitoring purposes, displaying submission histories and active assignments.

The final stage of prototype development is Testing and Evaluation. Evaluation is conducted at three levels: technical testing (performance, load, security), usability testing with representative users, and educational effectiveness testing through 3 assignments given to 9 students. After completing 3 assignments, students are asked to fill out questionnaires to measure several aspects, such as application functionality, UI/UX experience [21], and AI feedback quality. Figure 2 shows system architecture in this research.

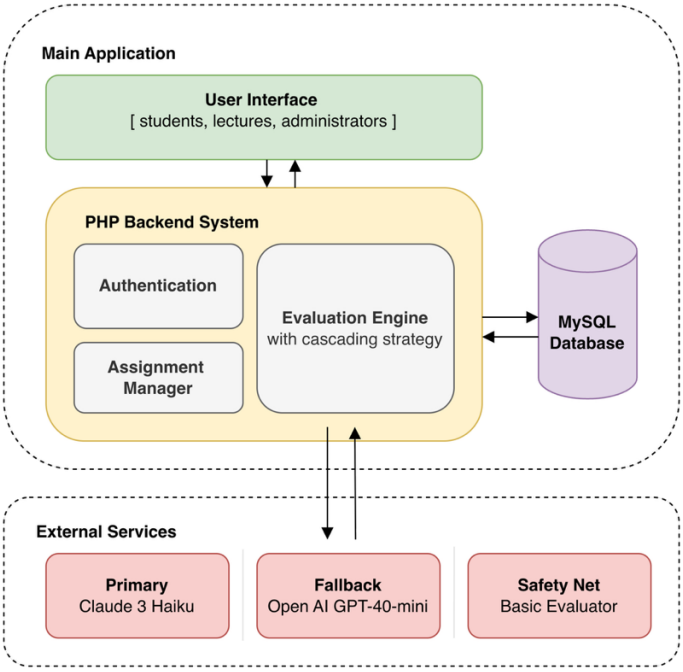

Figure 2. System Architecture

RESULTS AND DISCUSSION

System Development

The prototype website was developed at URL https://kodepy.site. The website's initial page provides 2 options for login and registration (Figure 3). Users can login with 3 role types: student (Figure 4), lecturer (Figure 5), and admin. Student role can access assignment pages, submit assignments, and receive/view AI feedback. A lecturer's role can create assignments, edit assignments, delete assignments, view submissions, and provide personal scores and feedback according to student submissions. Students who submit assignment answers are evaluated by AI, providing score responses, feedback, and suggestions. When lecturers access the view submission, they can see AI Feedback (score, comments, and suggestions from AI) and provide personal feedback and scores to students (Figure 6). Scores given by lecturers replace student scores obtained from AI (Figure 7).

System Evaluation Results

The final stage of prototype development is Testing and Evaluation. Evaluation is conducted at three levels: technical testing (performance, load, security), usability testing with representative users, and educational effectiveness testing through 3 assignments given to 9 students. After completing 3 assignments, students are asked to fill out questionnaires to measure several aspects such as application functionality, UI/UX experience, and AI feedback quality.

Participant Data and Submissions

Evaluation involved 9 active students in the Informatics Engineering Study Program at Ma Chung University taking the Programming course with a total of 39 successfully completed submissions. The distribution of participants and submission activities is shown in Table 3.

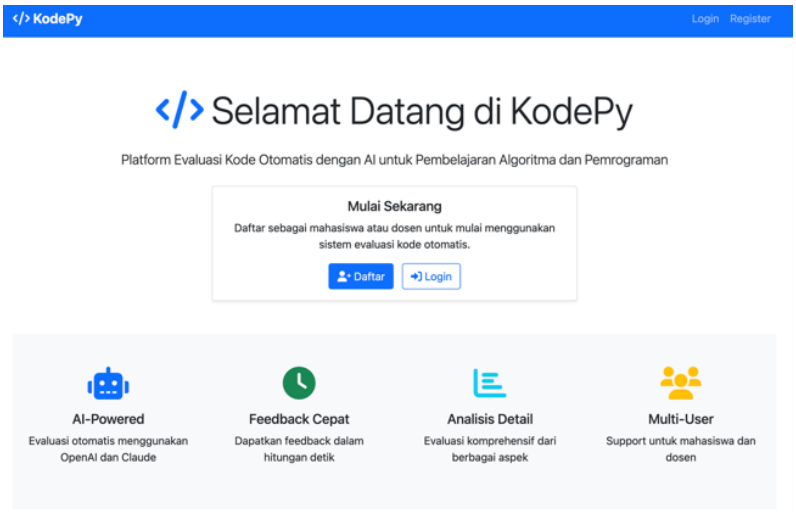

Figure 3. Homepage kodepy.site

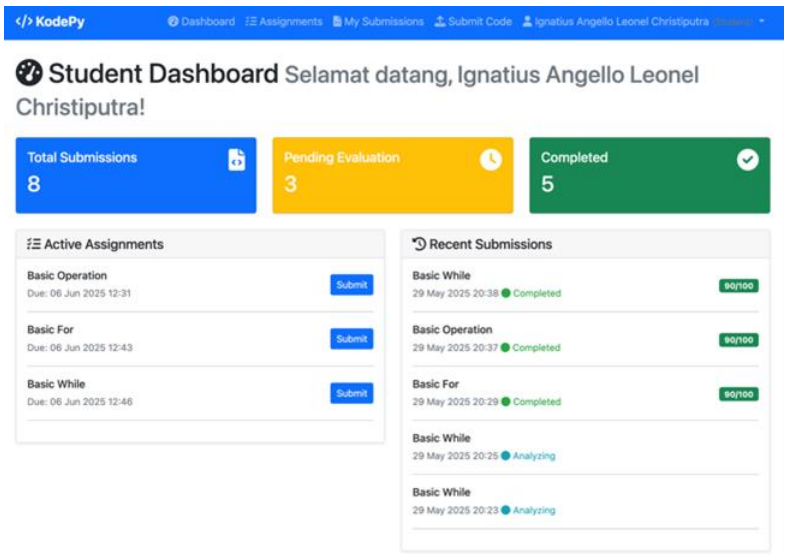

Figure 4. Student Dashboard

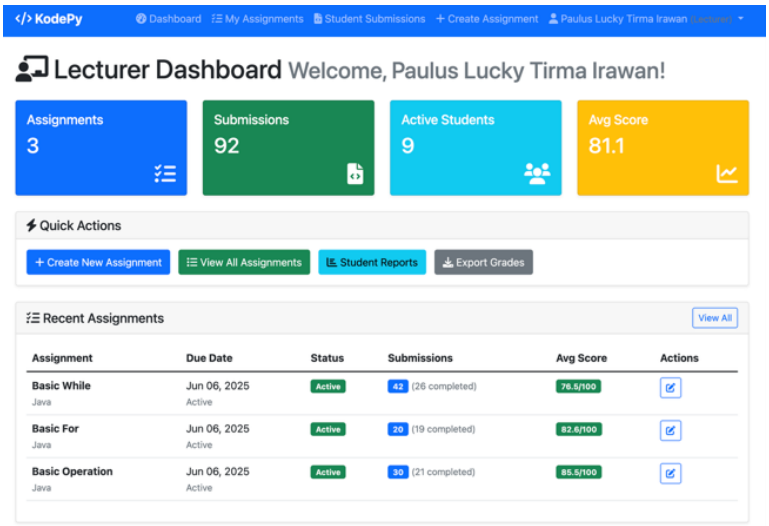

Figure 5. Lecturer Dashboard

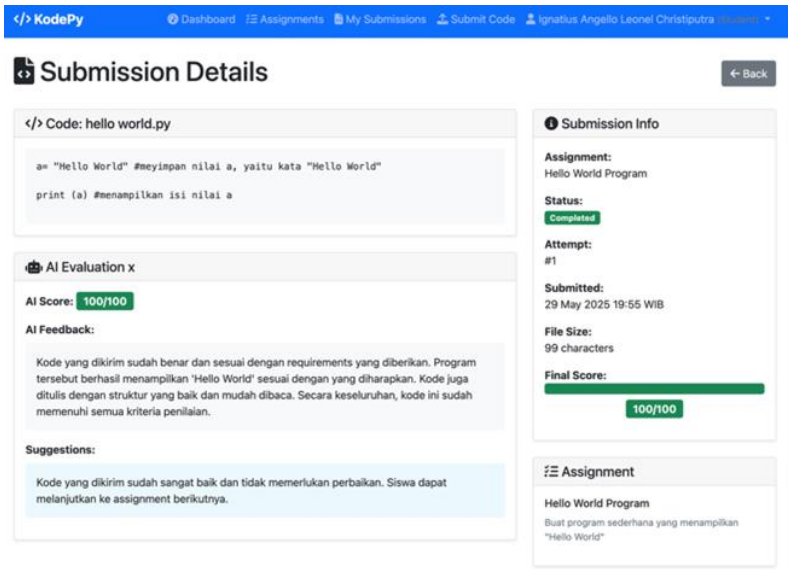

Figure 6. AI Evaluation Part

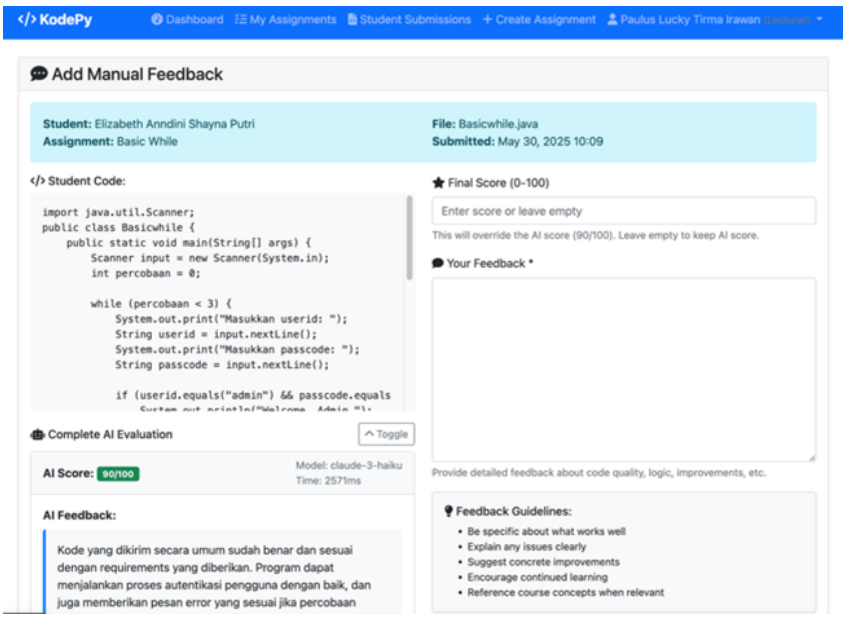

Figure 7. Lecturer Manual Feedback Part

Table 3. Participants and Submission Activities

Participants	Major	Submission
Student 1	Informatics Engineering	15
Student 2	Informatics Engineering	8
Student 3	Informatics Engineering	2
Student 4	Informatics Engineering	8
Student 5	Informatics Engineering	6
Student 6	Informatics Engineering	17
Student 7	Informatics Engineering	6
Student 8	Informatics Engineering	14
Student 9	Informatics Engineering	4

Table 4. Score Distribution Analysis

Assignment	>=90	80-89	60-79	<60	Average
#1	45%	25%	25%	5%	82.3
#2	35%	30%	35%	0%	84.7
#3	40%	35%	25%	0%	87.1

The system used a cascading AI evaluation strategy with priority order: Claude 3 Haiku as primary service, OpenAI GPT-4o-mini as fallback, and Basic Evaluator as final safety net. A structured evaluation prompt that is context-aware includes an assignment description for the relevant evaluation. Prompt is designed with multi-criteria evaluation using five criteria, including Syntax Correctness, Logic Correctness, Code Structure, Best Practices, and Efficiency.

AI Evaluation Score Distribution Analysis

To further analyze the results, the Claude AI system evaluated each submission using a multi-criteria framework. The scores were weighted: Syntax Correctness (25%), Logic Correctness (30%), Code Structure (20%), Best Practices (15%), and Efficiency (10%). The system automatically calculated composite scores from 0 to 100 based on these criteria. Table 4 shows the resulting score distribution analysis. Assignment #3 shows the highest average score (87.1), indicating students experienced improved programming concept understanding through the iterative learning process from previous assignments. No scores below 60 in the final assignment demonstrate AI feedback effectiveness in helping students correct their errors.

Claude AI Feedback Quality Analysis

Feedback quality evaluation was conducted through content and structure analysis of feedback generated by Claude AI. The system successfully provided comprehensive feedback covering syntax, logic, and programming best practices (Table 5). Claude AI demonstrated the capability of identifying syntax errors and providing program logic improvement suggestions. Both aspects are main problems for students learning to program. Generated feedback uses natural Indonesian language easily understood by students, with specific and actionable code examples. 100% evaluation success rate demonstrates Claude AI API integration stability with minimal technical failures. Average response time was 2.8 seconds with zero system failures or timeouts. The average API cost per evaluation was $0.02, with a total cost for 39 evaluations: $0.78.

User Satisfaction Survey Results

Survey involved 8 student respondents from Informatics Engineering Study Program at Ma Chung University. Respondents had experience using the system with varying submission numbers, ranging from 3 to 30 submissions.

Table 5. Feedback Category Generate by Claude AI

Feedback Category	Frequency
Syntax Correction	85%
Logic Improvement	75%
Code Structure	60%
Best practices	45%
Performance tips	25%

The ease of use aspect received positive responses from students. Most respondents agreed that the system interface is easy to understand and use, with intuitive navigation. The code file upload process was considered straightforward and user-friendly. However, some respondents provided feedback, such as suggestions to enable direct coding on the website without file upload requirements. AI feedback quality received positive responses from respondents. AI feedback was considered very helpful in identifying syntax errors and program logic. Student engagement pattern analysis reveals students made an average of 3.2 attempts per assignment, showing an active iterative learning process. Students made improvements based on AI feedback and resubmitted until achieving satisfactory scores.

Discussion

The KodePy system evaluation results demonstrate that the prototype successfully meets research objectives by providing high-quality automatic feedback. The system proves capable of supporting the programming learning process for students, evident from the high engagement rate and consistent improvement rate. The cascading AI evaluation strategy implementation provides system reliability through fallback mechanisms when primary services are unavailable. Cost optimization is achieved through cost-effective models like gpt-4o-mini, while comprehensive coverage ensures every submission receives evaluation. Multi-criteria evaluation with clear weights provides a comprehensive assessment covering not only syntax and logic correctness but also code structure, best practices, and efficiency aspects. This holistic approach aligns with programming education best practices, emphasizing code quality beyond mere functionality.

The progressive score improvement from the first assignment (82.3) to the third assignment (87.1), measured against a maximum target score of 100, indicates effective iterative learning supported by AI feedback. Students demonstrate the ability to learn from this feedback and apply concrete improvements in subsequent assignments. Furthermore, the high engagement rate, with an average of 3.2 attempts per assignment, shows that students actively utilize the system to incrementally approach the maximum score and achieve continuous learning improvement. AI feedback quality analysis shows strength in syntax correction (85%) and logic improvement (75%), addressing primary challenges faced by novice programmers. Natural language feedback in Indonesian enhances comprehension and applicability for local students. Technical performance with a 2.8-second average response time and 100% success rate demonstrates system reliability for educational use. Cost efficiency at $0.02 per evaluation makes the system economically viable for educational institutions.

While the initial results demonstrate a positive trend in score improvement and high user engagement, the authors acknowledge the limitations regarding data adequacy in this pilot phase. The evaluation involved a constrained sample size of 9 students and 39 submissions. Therefore, the current findings should be interpreted as preliminary indicators of technical feasibility and potential pedagogical benefits, rather than definitive proof of widespread educational effectiveness. The lack of broader inferential statistical modeling highlights the need for cautious generalization.

CONCLUSIONS AND SUGGESTIONS

This research successfully developed and evaluated the KodePy prototype as an intelligent automatic feedback system for algorithm and programming education, demonstrating strong technical performance, pedagogical value, and economic feasibility. The platform achieved an average response time of 2.8 seconds with a 100% evaluation

success rate across 39 submissions from 9 students, while maintaining a low operational cost of only $0.02 per evaluation, indicating strong scalability for educational institutions. The multi-criteria assessment framework effectively evaluated student work across five dimensions: Syntax Correctness (25%), Logic Correctness (30%), Code Structure (20%), Best Practices (15%), and Efficiency (10%), enabling more comprehensive formative assessment beyond conventional binary grading. Learning analytics showed clear progress, with average scores increasing from 82.3 in Assignment #1 to 87.1 in Assignment #3, while the proportion of students scoring below 60 declined from 5% to 0%, highlighting the system's support for struggling learners. Student engagement patterns revealed active iterative learning, with an average of 3.2 submission attempts per task, while AI feedback effectively addressed novice programming challenges, particularly syntax correction (85%), logic refinement (75%), and code organization (60%). User survey data confirmed high usability and trust in AI-generated feedback, achieving an overall acceptance rate of 90%. Beyond implementation, this study contributes conceptually through a Cascading AI Evaluation Strategy that addresses the reliability limitations of single-LLM grading systems and advances automated assessment toward holistic and formative evaluation. Although inferential statistical testing was limited by the pilot-scale sample size, the descriptive findings strongly validate the system's technical and educational feasibility. Future large-scale studies should incorporate rigorous statistical modeling to quantify learning gains and long-term pedagogical impact.

ACKNOWLEDGMENTS

The authors thank Ma Chung University for supporting this research and all students who participated in the system evaluation process.

REFERENCES

[1] Hao, Q., Smith IV, D.H., Ding, L., Ko, A., Ottaway, C., Wilson, J., Arakawa, K.H., Turcan, A., Poehlman, T. and Greer, T., "Towards understanding the effective design of automated formative feedback for programming assignments," Computer Science Education, vol. 32, no. 1, pp. 105-127, 2022.

[2] M. M. Fakhri, et. al., "Barriers to Effective Learning: Examining the Influence of Delayed Feedback on Student Engagement and Problem Solving Skills in Ubiquitous Learning Programming," J. Appl. Sci. Eng. Tech. Educ., vol. 6, no. 1, pp. 69-79, Jul. 2024.

[3] Z. Zhou, S. Wang, dan Y. Qian, "Learning From Errors: Exploring the Effectiveness of Enhanced Error Messages in Learning to Program," Front. Psychol., vol. 12, Nov. 2021

[4] Barra, E., López-Pernas, S., Alonso, Á., Sánchez-Rada, J.F., Gordillo, A. and Quemada, J., "Automated assessment in programming courses: A case study during the COVID-19 era," Sustainability, vol. 12, no. 18, Art. no. 7451, 2020.

[5] Azaiz, I., Kiesler, N. and Strickroth, S., "Feedback-generation for programming exercises with gpt-4," in Proc. 2024 Innovation and Technology in Computer Science Education V. 1, pp. 31-37, 2024.

[6] H. Atun dan E. Usta, "The effects of programming education planned with TPACK framework on learning outcomes," Participatory Educational Research, vol. 10, no. 3, pp. 67-82, 2023.

[7] M. Beege, et. al., "Learning programming from erroneous worked-examples. Which type of error is beneficial for learning?", Learning and Instruction, vol. 75, p. 101497, 2021.

[8] H.-C. Ling dan H.-S. Chiang, "Learning Performance in Adaptive Learning Systems: A

Case Study of Web Programming Learning Recommendations," Frontiers in Psychology, vol. 13, p. 770637, Jan. 2022.

[9] M. Kwak, J. Jenkins, dan J. Kim, "Adaptive programming language learning system based on generative AI," International Journal of Intelligent Systems, vol. 24, no. 3, pp. 222-231, 2023.

[10] Croft, D. and England, M., "Computing with CodeRunner at Coventry University: Automated summative assessment of Python and C++ code," in Proc. 4th Conference on Computing Education Practice, pp. 1-4, Jan. 2020.

[11] Vetrivel, S.C., Arun, V.P., Ambikapathi, R. and Saravanan, T.P., "Automated Grading Systems: Enhancing Efficiency and Consistency in Student Assessments," in Adopting Artificial Intelligence Tools in Higher Education, CRC Press, pp. 41-61, 2025.

[12] N. Scholz, et.al., "Partnering with AI: A Pedagogical Feedback System for LLM Integration into Programming Education," arXiv preprint arXiv:2507.00406, Jul. 2025.

[13] Messer, M., Brown, N.C., Kölling, M. and Shi, M., 2024. Automated grading and feedback tools for programming education: A systematic review. ACM Transactions on Computing Education, 24(1), pp.1-43.

[14] Souza, D.M., Felizardo, K.R. and Barbosa, E.F., "A systematic literature review of assessment tools for programming assignments," Computer Applications in Engineering Education, vol. 24, no. 6, pp. 752-774, 2016.

[15] Paiva, J.C., Leal, J.P. and Figueira, Á., "Automated assessment in computer science education: A state-of-the-art review," ACM Transactions on Computing Education, vol. 22, no. 3, pp. 1-40, 2022.

[16] Mertler, C.A., Introduction to educational research. Sage publications, 2024.

[17] L. Afriani, "Understanding the design of research and development methods in the field of education," IJESS Int. J. Educ. Soc. Sci., vol. 6, no. 1, pp. 1-5, 2025.

[18] Ahmad, S.I., Rana, T. and Maqbool, A., "A model-driven framework for the development of MVC-based (Web) application," Arabian Journal for Science and Engineering, vol. 47, no. 2, pp. 1733-1747, 2022.

[19] S. Suarman, M. M. Afandi, dan M. K. Sari, "A Needs Analysis for Research-Based TPACK Models," Alishlah J. Stud. Islam., vol. 12, no. 1, pp. 45-60, Jan. 2025.

[20] A. Ahmed et al., "Design and implementation of a responsive web-based system for controlling the financial budget of universities," Journal of Technology and Informatics (JoTI), vol. 5, no. 1, pp. 1-7, 2023.

[21] A. Nugroho et al., "UI/UX design of a web-based student organizations system using the design thinking method approach," Journal of Technology and Informatics (JoTI), vol. 7, no. 1, pp. 24-38, 2025