Article

Teaching Reform of "Java Web Application Development" under the Mode of "AI Co-Creation + Competition and Teaching Integration"

Yuxiang Hou [1,*], Weikai Ye [1] and Zhihui Xu [1]

[1] Department of Computing, Guangdong University of Science and Technology, Dongguan, China
* Correspondence: Yuxiang Hou, Department of Computing, Guangdong University of Science and Technology, Dongguan, China

Abstract: This study explores the teaching reform of the course "Java Web Application Development" under a blended model that combines AI co-creation with the integration of competition and teaching. With the rapid development of artificial intelligence and its widespread adoption in software engineering, traditional lecture-centered approaches are increasingly unable to meet the needs of applied talent training. In response, this paper introduces AI-assisted tools into the full teaching process, including requirements analysis, interface design, coding, debugging, and deployment, so as to support human–machine collaboration and improve students' problem-solving abilities. At the same time, discipline competitions and project-based tasks are embedded into the curriculum to form a closed loop of "learning–practice–competition–reflection". The reform focuses on optimizing teaching objectives, reconstructing knowledge modules, and designing progressive practical projects that align with real enterprise scenarios. Through the combination of formative assessment, process evaluation, and competition results, a diversified evaluation system is established to better reflect students' comprehensive competencies. Teaching practice shows that the AI co-creation and competition–teaching integration model can significantly enhance students' learning engagement, programming literacy, engineering awareness, and innovative thinking, while also improving the effectiveness of Java Web application development teaching. The research provides a useful reference for the reform of similar programming and software development courses in higher education.

Keywords: java web; AI co-creation; teaching reform; human–computer collaboration; programming education

Received: 27 January 2026
Revised: 14 March 2026
Accepted: 29 March 2026
Published: 05 April 2026

1. Introduction

1.1. Research Background and Question Proposal

Under the context of new engineering construction, the "Java Web Application Development" course faces three structural challenges: the disparity between the rapid evolution of technological ecosystems and the delayed updates of textbooks, resulting in a disconnect between classroom learning and enterprise application [1]. The teaching approach emphasizes grammar explanations and verification experiments, leaving students without sufficient engineering training in complete projects. The standardized teaching system overlooks the fundamental differences among students, suppressing the demand for personalized learning over an extended period. Simultaneously, generative AI programming assistants are transforming the software development paradigm, and the increasing demand for "AI collaborative development capabilities" in corporate recruitment is compelling higher education to reassess the teaching objectives of programming courses. Additionally, the scale of computer discipline competitions continues to grow. The emphasis on solving complex problems, fostering innovative thinking, and promoting teamwork aligns well with addressing the deficiencies of

traditional classroom teaching. However, current research often treats the integration of AI tool applications and competition-based education as separate issues, lacking a systematic integration framework. This highlights the urgent need to explore a cohesive approach to unify these aspects effectively [2].

1.2. Definition of Core Concepts

"AI Co-creation" specifically refers to a form of human-computer collaboration teaching activity, positioning generative AI as an "equal collaborator" rather than a simple tool. Teachers and students form a triangular interaction with AI: teachers design higher-order cognitive tasks and guide AI use boundaries, students drive AI to generate code drafts, debug and optimize architecture schemes through accurate prompts, and ultimately humans dominate creative decision-making and quality control. This concept emphasizes the collaborative and bidirectional construction of knowledge production, and its educational value lies not only in the improvement of efficiency but also in the exposure of the thinking process and the promotion of metacognitive development through "human-machine dialogue." The "integration of competition and teaching" refers to the deep embedding of the standard system, project resources, evaluation mechanism, and incentive mechanism of subject competition into the entire process of course teaching, achieving the teaching organization mode of "integration of class and competition." Its connotation includes three levels: reconstructing the curriculum learning objectives with the industry competence standards mapped by the competition; transforming the real questions of previous competitions and the real needs of enterprises into stepwise teaching projects; and introducing the diversified evaluation dimensions of the competition to replace single assessments [3]. It should be emphasized that the integration of competition and teaching is not utility-based exam-oriented training but aims to stimulate learning drive and cultivate engineering thinking through the competition mechanism.

1.3. Research Purpose and Significance

This study aims to construct a course teaching reform model driven by the dual approach of "AI co-creation and competition-teaching integration." The specific objectives include designing a teaching framework that integrates the AI technology chain with the competition ability chain, developing operational strategies and toolkits, and providing a theoretical basis for subsequent teaching experiments to validate the model's effectiveness. At the theoretical level, the "double helix structure model" and the "human-computer division boundary" theory proposed in this study offer a conceptual framework for transforming programming education paradigms in the era of generative AI. These theories also extend the application scope of human-computer collaborative learning theory within programming education [4]. At the practical level, the research outcomes can directly inform curriculum reform practices in application-oriented colleges and universities, provide replicable solutions for integrating artificial intelligence into higher education, and guide students in balancing technical convenience with skill development to mitigate the risk of cognitive inertia caused by over-reliance on technology.

1.4. Research Status and Theoretical Basis at Home and Abroad

1.4.1. Review of Research Status

International academic research on the educational application of AI programming tools indicates that the use of tools such as Copilot can lower the entry threshold for programming but may hinder the development of algorithmic thinking for beginners. Recent studies have shifted focus toward fostering "AI collaboration ability," suggesting that "Prompt Engineering Literacy" should be incorporated into computer science curricula. Regarding the integration of competition and teaching, international competitions such as ACM-ICPC have long served as teaching resources for algorithm courses. However, systematic research has predominantly concentrated on top-tier

universities, with limited attention given to the broader applicability for application-oriented institutions. Domestic research exhibits a policy-driven approach, with the introduction of generative AI into classrooms emerging as a prominent topic. Researchers have systematically examined the ethical risks and strategies for addressing challenges in AI-assisted programming education, proposing a "double-teacher classroom" model based on large language models. In the domain of competition and teaching integration, studies primarily focus on analyzing competition mechanisms or individual course transformation cases, while research on the systematic integration of AI tools and competition mechanisms remains underexplored [5]. Existing literature reveals several shortcomings: AI is often treated as an efficiency tool rather than a cognitive partner; research on competition and teaching integration emphasizes competition outcomes without adequately addressing the micro-level mechanisms of teaching process reconstruction. Additionally, there is a notable lack of specialized research on applied courses related to web development.

1.4.2. Theoretical Basis

Constructivism learning theory emphasizes that knowledge is not passively received but actively constructed. In the "AI co-creation" scenario, AI acts as a "more capable peer" within the framework of the "zone of nearest development" theory, assisting students in surpassing their ability boundaries through immediate feedback and scaffolding support. However, the code generated by AI requires students to critically evaluate and adapt it. This process fosters deep understanding and metacognitive development. Situated learning theory underpins the integration of competition and teaching by framing subject competition as a highly situated community of practice [6, 7]. Embedding competition into the curriculum positions students in "legitimate marginal participation," gradually advancing them toward the core of the community of practice. Human-computer cooperative cognition theory highlights the complementarity between humans and AI in cognitive tasks and advocates for a collaboration model of "human-led creativity and AI-assisted implementation." This approach leverages the efficiency of AI while preserving students' higher-order thinking skills, mitigating the risk of skill degradation that could arise from over-reliance on AI.

2. The Design of Teaching Mode of "AI Co-creation + Competition and Teaching Integration"

2.1. Pattern Architecture Design

In this study, the "double helix structure model" is proposed as the core architecture of the teaching model, with the AI technology chain and the competition ability chain serving as two intertwined main chains. The AI technology chain encompasses the full life cycle technical support, including requirements analysis, design, coding, testing, and deployment. The competition ability chain maps to problem analysis, system design, engineering implementation, innovation application, and team collaboration as core competence dimensions [8]. The interleaving points of the two main chains correspond to specific teaching modules, where AI alleviates lower-order cognitive loads, enabling students to concentrate their cognitive resources on cultivating higher-order abilities. The model architecture also incorporates three supporting systems: the AI governance system, which establishes usage norms and fosters prompt engineering skills; the project ecosystem, which integrates a competition question library, enterprise case library, and open-source component library; and the evaluation support system, which develops expressive evaluation tasks, process recording tools, and capability development profiles.

2.2. Reconstruction of Teaching Elements

The role of teachers transitions from knowledge transmitters to guides in AI collaboration and competition coaching, requiring mastery of three new abilities: prompting engineering design, AI code review, and competition guidance. Their focus

shifts from explaining syntax details to designing high-level challenge tasks and organizing code review sessions. Students evolve from passive recipients to active creators and prompting engineers, with essential skills including articulating requirements to AI, critically evaluating AI outputs, guiding AI through iterative improvement proposals, and managing competition-level time constraints and quality expectations as "contestants." The learning resource system has been dynamically restructured, replacing fixed textbooks with flexible resource packages. These include personalized learning paths generated by AI, exemplary works from previous competitions, interactive AI programming assistants, and real-time updated technical radars. Resource organization adheres to a "competition-driven" framework [9]. Course assignments are evaluated based on competition standards, participation in competitions is integrated into assessments, and AI collaboration archives document interaction histories for skill evaluation.

2.3. Curriculum Module Reorganization

The curriculum is reorganized into three progressive levels. The base layer focuses on "AI-assisted grammar and framework speed learning," accounting for 30% of the class hours. The "reverse engineering" strategy is employed to directly present the complete project, guide students in using AI to explain the project structure, establish technical panoramic cognition through "deconstruction-reconstruction," and set up an "AI-disabled period" to enforce handwriting of core code, ensuring foundational skills. The "competition-driven project practice" is implemented at the progressive level, accounting for 50% of the class hours. A selection of 10-15 competition-driven projects forms the project ladder, with each project following a structured process: requirements analysis, scheme defense, development sprint, code review, and demonstration review. This approach emphasizes cultivating the ability for rapid comprehension and efficient collaboration under pressure. The innovation layer involves "complete Web application development with AI collaboration," accounting for 20% of the class hours. Students form teams to address real-world needs, completing the entire development process. AI plays a significant role in competitive product analysis, document generation, and deployment optimization [9, 10]. The final outcomes are evaluated through competition selection, highlighting the integration of innovative thinking and engineering integrity.

3. Teaching Implementation Path and Strategy

3.1. Implementation Strategies for AI Co-Creation

The hierarchical prompt design strategy enhances prompt engineering capabilities, progressing from basic usage to advanced mastery. The first layer, "descriptive prompt," provides a structured framework comprising four elements: role, task, constraint, and output format. The second layer, conversational prompts, facilitates iterative optimization through multiple rounds of refinement. The third layer, "metacognitive prompts," emphasizes reflection on foundational design principles and comparative analysis of AI outputs. The AI code review mechanism establishes a three-tier quality assurance process: initial inspection by AI using static analysis tools to identify vulnerabilities, peer evaluation through structured review tables, and final review by teachers focusing on architectural soundness and innovation. The human-machine division of labor strategy mitigates over-reliance by enforcing clear rules: AI is restricted to ensure higher-order thinking during requirements analysis and architecture design stages, the core algorithm must be handwritten and undergo review, and code logic must be explained during the final defense. AI-generated boilerplate code is encouraged, while testing and documentation tasks can be fully supported by AI.

3.2. Implementation Strategy of Integration of Competition and Education

The standard curriculum strategy for the competition transforms scoring rules into curriculum learning objectives and evaluation scales, encompassing dimensions such as

functionality, reliability, performance efficiency, innovation, and user experience. These dimensions correspond to requirements realization, exception handling, concurrent handling, technical highlights, interaction design, and other indicators within the course. The project transformation strategy for real-world questions establishes a sustainable and updated project library, which includes processes such as question screening, situation reconstruction, hierarchical disassembly, and resource matching [11]. This project library enables personalized selection based on technical and business dimensions. Final course projects are utilized as entries for school-level selection through a competition examination mechanism. Performance evaluation employs a dual-track system combining course performance with competition bonus points. Outstanding participants can receive excellent evaluations and innovation credits, while incomplete projects may still earn competition certificates, serving as endorsements for career advancement.

3.3. Teaching Process Design

The pre-class stage utilizes AI to generate personalized preview tasks, deliver differentiated materials based on pre-knowledge diagnostics, and incorporate an "AI dialogue challenge" to foster independent inquiry habits. During the class, a "double-teacher classroom" approach combined with a competition-style, time-limited development framework is implemented. The human teacher focuses on demand interpretation and guiding critical thinking, while the AI assistant handles instant question answering and code generation. The time is structured as "20 minutes of explanation, 40 minutes of practice, and 20 minutes of review," with a "competition simulation day" scheduled weekly for pressure testing. In the post-class stage, emphasis is placed on AI-driven code review. Students submit development logs to document AI usage, challenges encountered, and reflections. Teachers provide personalized feedback, and outstanding projects are added to the project library for iterative updates.

4. Construction of Teaching Evaluation System

4.1. Multiple Evaluation Index System

The evaluation system establishes three-dimensional indicators encompassing AI collaboration ability, engineering practice ability, and innovative thinking. The AI collaboration capability dimensions include the quality of hints (role setting, task description, context adequacy), critical assessment of AI output (security vulnerability checking, performance evaluation, boundary verification), and human-machine collaboration fluency (call timing, error identification, requirements refinement). The engineering practice ability dimension includes code standardization (Git commit specification, code reuse, exception handling), system performance (response time, query efficiency, memory optimization), and deployment operation and maintenance capabilities (containerization, CI/CD, monitoring log). The creative thinking dimension includes demand insight (implicit demand identification, abnormal process consideration, competitive product analysis), architecture design uniqueness (technology innovation, emerging technology introduction, DDD idea), and technology selection rationality (maintenance status, community activity, learning cost). These three-dimensional indicators form a hierarchical structure of "base-support-leading," enabling students to develop diverse competencies [12, 13].

4.2. Evaluation Tools and Implementation Methods

Evaluation instruments are categorized into three types. Performance assessment tasks spanned the entire semester, including AI dialogue challenges in week 2, code walkthrough sessions in weeks 4, 8, and 12, architecture design defenses in weeks 6 and 14, time-limited development sprints in weeks 5, 9, and 13, and course work reviews in week 16. Transparent scoring rules were provided in advance for each task. Procedural data collection tools included AI usage logs, which recorded scenarios, satisfaction and failure experiences, and improvement plans; Git commit reflections, which explained

actions, reasoning, and reference resources; and an ability self-rating scale using a 9-dimensional Likert 5-level score, conducted at the beginning, middle, and end of the semester. The comprehensive evaluation portfolio compiled work evidence, process evidence, reflection evidence, and external evidence, utilizing an overall performance rating system instead of a hundred-mark system [14]. Evaluation was distributed among students' self-evaluation (30%), peer evaluation (20%), teacher evaluation (30%), and assessments by enterprise engineers or competition judges (20%).

4.3. How to Use the Evaluation Results

The evaluation results were utilized for tracking individual development and enhancing teaching practices. At the individual level, an "ability radar chart" was generated to illustrate students' development trajectories without incorporating class averages. Based on this, students formulated personal development plans, while teachers provided tailored resource recommendations. If challenges in identifying AI code vulnerabilities were commonly observed, specialized training sessions were introduced. Similarly, if projects encountered obstacles, teaching priorities were adjusted, with formative cycles relying on classroom observations rather than extensive sample data. At the course level, the "Curriculum evaluation implementation report" documented the ease of evaluating indicators, the effectiveness of tools, and iterative suggestions for improvement [15].

5. Reform Implementation Plan and Expected Effect Deduction

5.1. Analysis of Conditions for Reform Implementation

The implementation conditions are evaluated across four dimensions. Regarding faculty, at least two educators with enterprise experience and expertise in competition guidance are required [9]. Part-time coaches can be recruited or promoted through enterprise practice flow stations. From a technological perspective, AI programming assistants and online evaluation systems must be deployed, with initial investments allocated for software licensing and server resources. In terms of institutional frameworks, it is essential to coordinate the credit recognition scheme, incorporate alternative assessments for competition winners, and integrate evaluation metrics for AI collaboration capabilities.

5.2. Extrapolating Expected Effects

Based on the theoretical model and similar research evidence, the deduction effect indicates improvements in code standardization through the use of AI-generated specification templates. Regarding project completion, the integration of competition and education provides clear targets, enhancing expected delivery capabilities. For the development of higher-order abilities, the division of labor between humans and machines releases cognitive resources, aligning with cognitive load theory and optimizing system design performance. In terms of AI collaborative literacy, structured prompt training demonstrates that the foundational methods of prompt engineering can be effectively mastered.

5.3. Potential Risks and Countermeasures

Risk prediction and response: To address potential vulnerabilities caused by excessive reliance on AI, an "AI disabled period" is implemented for pure white-box training, with handwritten code included in assessments, accounting for at least 20%. Additionally, an early warning mechanism is established to monitor dependency levels [16, 17]. To mitigate the risk of utilitarian learning driven by competitive tendencies, multiple evaluation metrics such as technical debt index, open-source contribution scores, and social service certifications are introduced. Code appreciation lectures are conducted to promote engineering aesthetics. In cases where the cost of AI tools becomes prohibitive,

localized open-source solutions, such as CodeLlama, are developed, and an AI usage quota system is established to ensure equitable access to basic computing resources.

6. Pattern Application Example (Based on Competition questions)

Using the "library management system" as an example, which was a real question from a 2023 provincial competition in the web development category, the implementation scenario was analyzed. During the requirement analysis stage, students utilized AI to compare open-source projects and generate a comparison table, while instructors provided guidance to identify implicit constraints [18]. At this stage, AI training demand insight was not employed. In the architecture design phase, students independently created architecture diagrams, using AI solely to query technology compatibility information. During the code implementation phase, AI was permitted to generate boilerplate code, but the core business logic was required to be handwritten and reviewed, with AI usage logs submitted for evaluation. In the test deployment phase, AI assisted in generating test cases, while students supplemented these with boundary testing and used AI to write Dockerfiles, ensuring the configuration was thoroughly explained. This process illustrates the specific application of a human-machine division of labor framework.

7. Conclusions

The two-wheel drive mode of "AI co-creation + competition and teaching integration" developed in this study achieves the interweaving and integration of AI technology and competition capabilities through a double helix structure model. A three-dimensional evaluation system encompassing AI collaboration, engineering practice, and innovative thinking abilities has been designed, alongside implementation strategies such as hierarchical prompting design, human-machine task boundaries, and competition-standard curricula. This work provides both a theoretical framework and a practical operational scheme for reforming Java Web application development courses in the era of generative AI. Future efforts will assess its impact on enhancing students' advanced skill development through teaching experiments and investigate its adaptability to other programming courses.

Funding: This research was supported by the Project of Teaching Quality and Teaching Reform of Undergraduate Universities in Guangdong Province in 2025 (serial number: 338) and the Quality Project of Guangdong University of Science and Technology (project number: GKZLGC2025020).

References

1. C. Stephens, C. A. Fisher, and C. Seckman, "A Generative AI Virtual Teaching Assistant for Graduate Nursing Informatics Education: Design, Implementation, and Preliminary Outcomes," CIN: Computers, Informatics, Nursing, vol. 10-1097, 2025.
2. X. Chen and L. Wang, "Design and Implementation of an Intelligent Education Assistance System Based on Internet of Things and Artificial Intelligence," International Journal of High Speed Electronics and Systems, vol. 2540842, 2025.
3. V. Bhatt, M. Brahmbhatt, M. S. Malek, and R. Rajai, "AI-driven classrooms: building creative pedagogies through digital literacy and infrastructure empowerment an AI-TPACK approach," Universal Access in the Information Society, vol. 25, no. 2, pp. 35, 2026.
4. M. M. Hasan, J. B. Mirza, R. Paul, M. R. Hasan, A. Hassan, and M. A. Islam, "Human-AI Collaboration in Software Design: A Framework for Efficient Co-Creation," AIJMR-Advanced International Journal of Multidisciplinary Research, vol. 3, no. 1, 2025.
5. S. Picault, G. Niang, V. Sicard, B. Sorin-Dupont, S. Assié, and P. Ezanno, "Leveraging artificial intelligence and software engineering methods in epidemiology for the co-creation of decision-support tools based on mechanistic models," Preventive Veterinary Medicine, vol. 228, pp. 106233, 2024.
6. G. Vindigni, "Entrepreneurship: The Value-Added of Co-Creation Through Web," European Journal of Applied Sciences–Vol, vol. 11, no. 3, 2023.
7. X. Kong, Y. Liu, Y. Shi, J. Xu, and M. Liu, "Mechanism of public behavioral intention to use generative AI for folk story image co-creation," npj Heritage Science, vol. 14, no. 1, pp. 164, 2026.
8. M. Jonsson and J. Tholander, "Cracking the code: Co-coding with AI in creative programming education," in Proceedings of the 14th Conference on Creativity and Cognition, pp. 5-14, June 2022.

9. C. Ceccarini, A. Liçaj, E. Matteucci, and G. Delnevo, "From Data to Narrative: Visualizing Complex Phenomena through Human-AI Co-Creation," in CEUR WORKSHOP PROCEEDINGS, vol. 4072, pp. 23-32, 2025.
10. J. H. Rautell and J. Saikkonen, "Drupal platform development-A case study of Co-Creation," 2013.
11. Q. Zhang, "Exploring Value Creation in Human-AI Collaborative Art: An Affordance Perspective," Information Systems Frontiers, pp. 1-26, 2026.
12. I. Makhonko, "Design and development of a web-based prototype for human-AI collaboration: integration an AI participant to enhance creativity," 2025.
13. G. Zoni, "Human-AI co-creation: an interaction design model for human-AI collaboration in software development," 2024.
14. M. Assante, L. Candela, D. Castelli, R. Cirillo, G. Coro, A. Dell'Amico, ... and F. Sinibaldi, "Virtual research environments co-creation: The D4Science experience," Concurrency and Computation: Practice and Experience, vol. 35, no. 18, pp. e6925, 2023.
15. Y. Zhuang, Y. Ji, Z. Huang, Y. Shang, X. Li, and X. Su, "Exploring the Integration of Artificial Intelligence and Programming Education: A Practice-Oriented Study on the Development of Computational Thinking Based on Human-AI Co-Creation," in 2025 IEEE 5th International Conference on Software Engineering and Artificial Intelligence (SEAI), pp. 325-329, June 2025.
16. M. Gomez Lindblom, "Empowering Web Editors with Generative AI: Creating a Tool for Efficient Search Engine Optimization within a Content Management System," 2025.
17. M. Hassany, P. Brusilovsky, J. Ke, K. Akhuseyinoglu, and A. B. L. Narayanan, "Human-AI co-creation of worked examples for programming classes," arXiv preprint arXiv:2402.16235, 2024.
18. S. Freese, "AI in Co-Creation: The usability and impact of AI tools for co-creation in participatory design to generate innovative and user-centric design solutions," 2023.